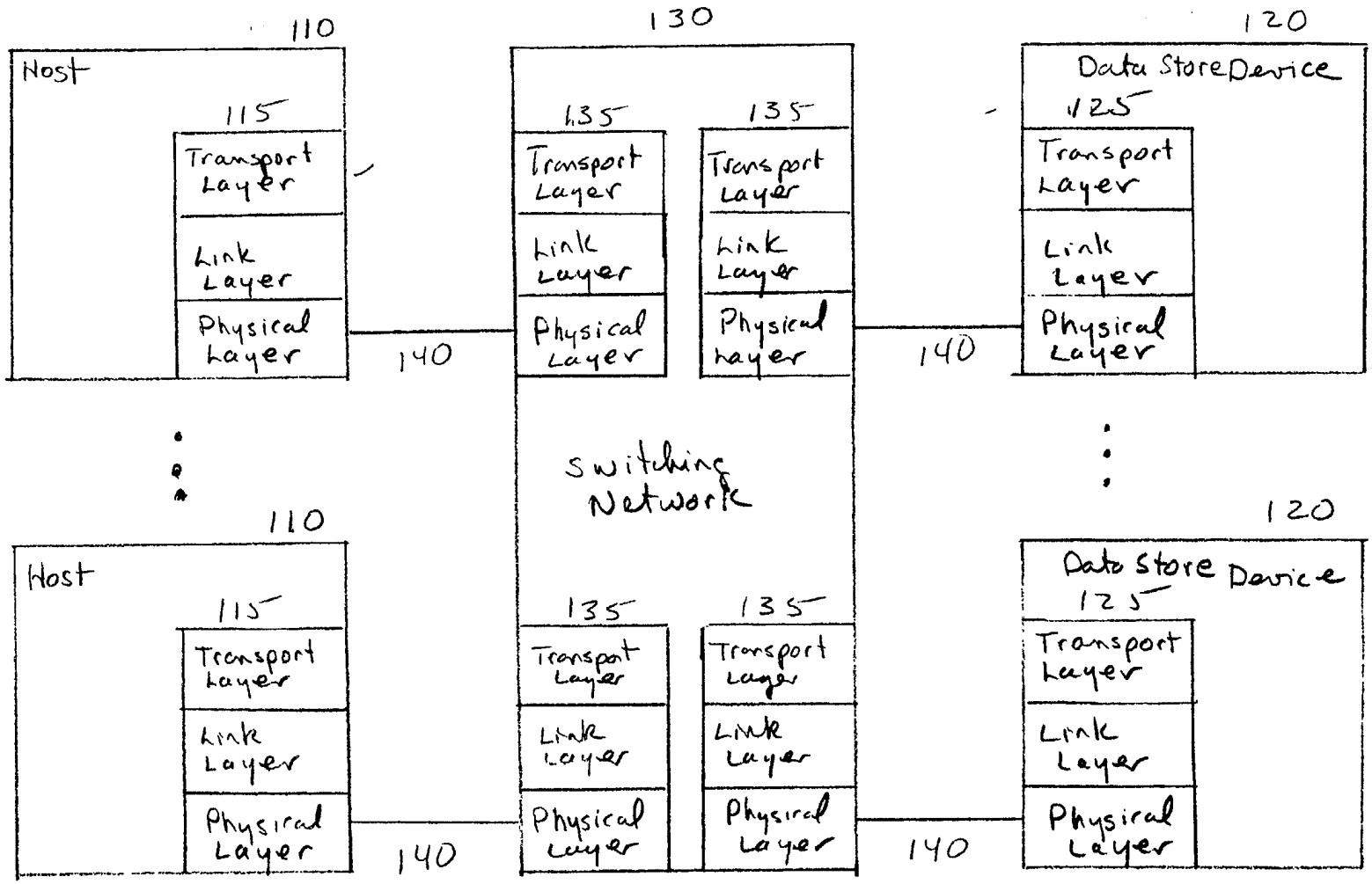


Fig 1

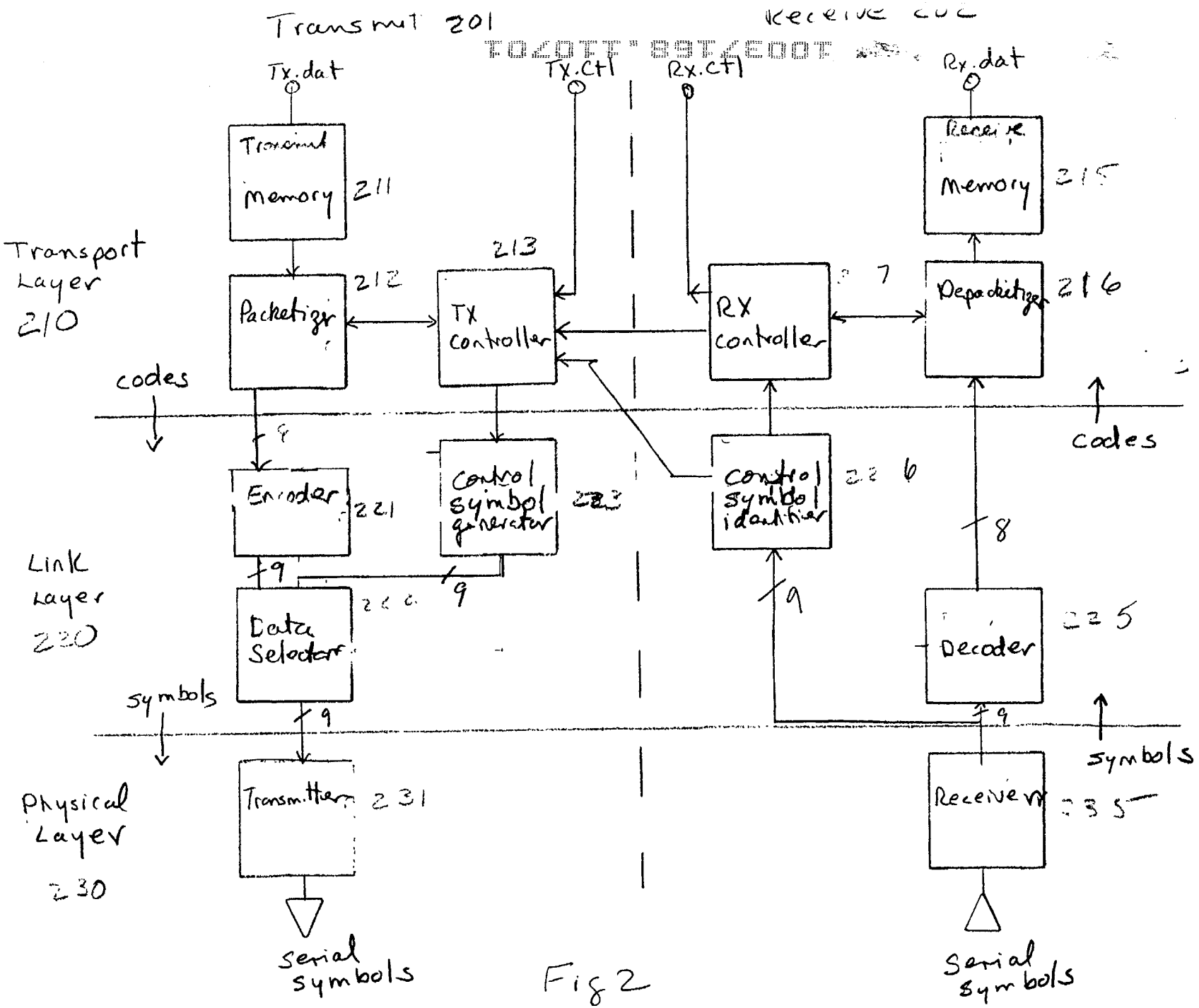

Transmit 201
Receive
Tx.dat
TX.Ctl
RX.Ctl
Rx.dat
Transmit memory 211
Receive memory 215
Transport Layer 210
213
Packetizer 212
TX controller
RX controller
217
Depacketizer 216
codes
codes
8
Encoder 221
control symbol generator 223
control symbol identifier 226
Link Layer 220
9
9
9
8
Data Selector 222
Decoder 225
symbols
symbols
9
9
Physical Layer 230
Transmitter 231
Receiver 235
serial symbols
Serial symbols

Fig 2

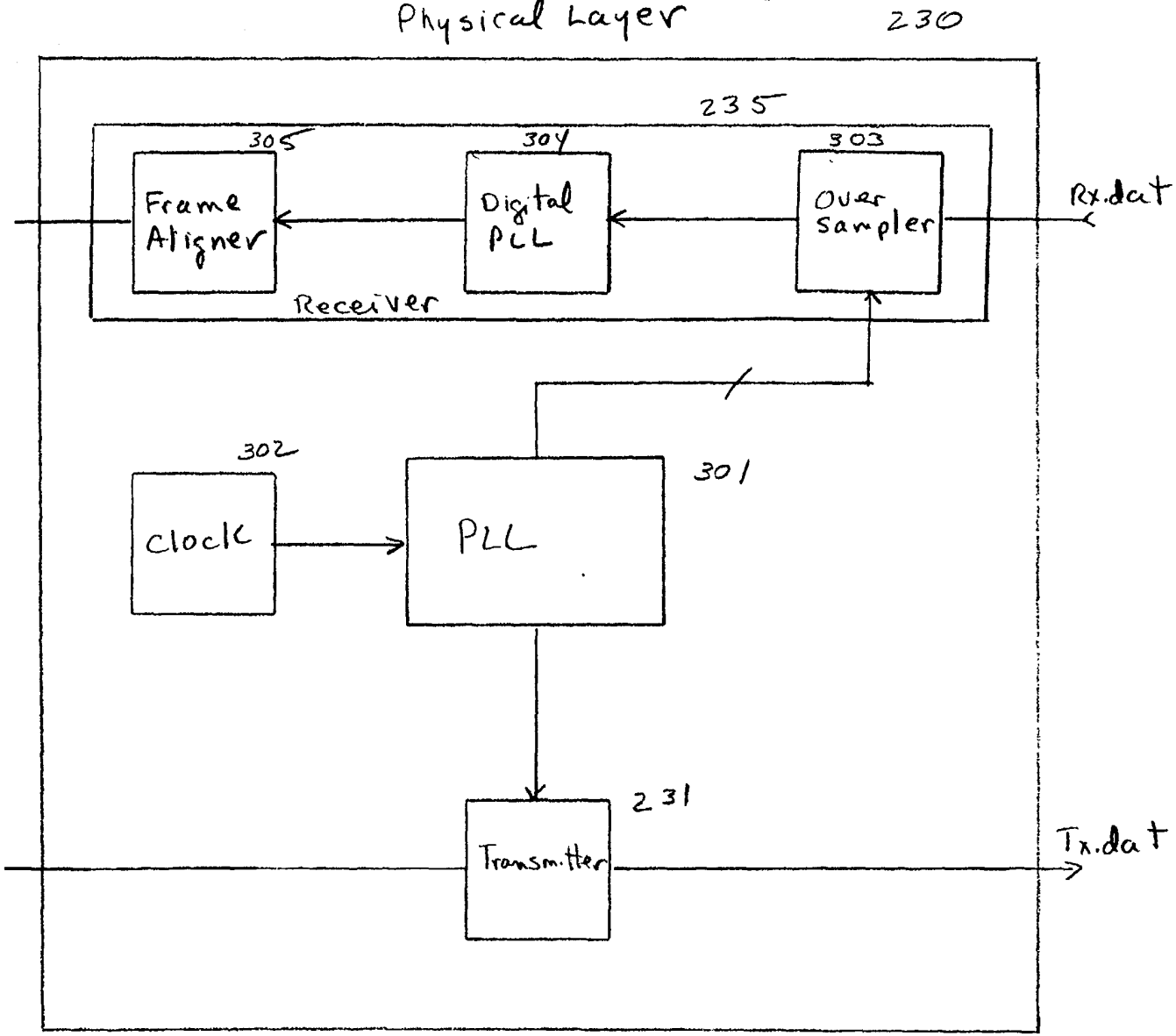


Fig 3

Packet 400

large →
medium →
small →

Reserved 1 411		Reserved 2 412		410 header
Destination Address 413		Source Address 414		
Type 415	Protocol 416	Class-specific control 417	Transaction Index 418	
Payload				420

Fig 4

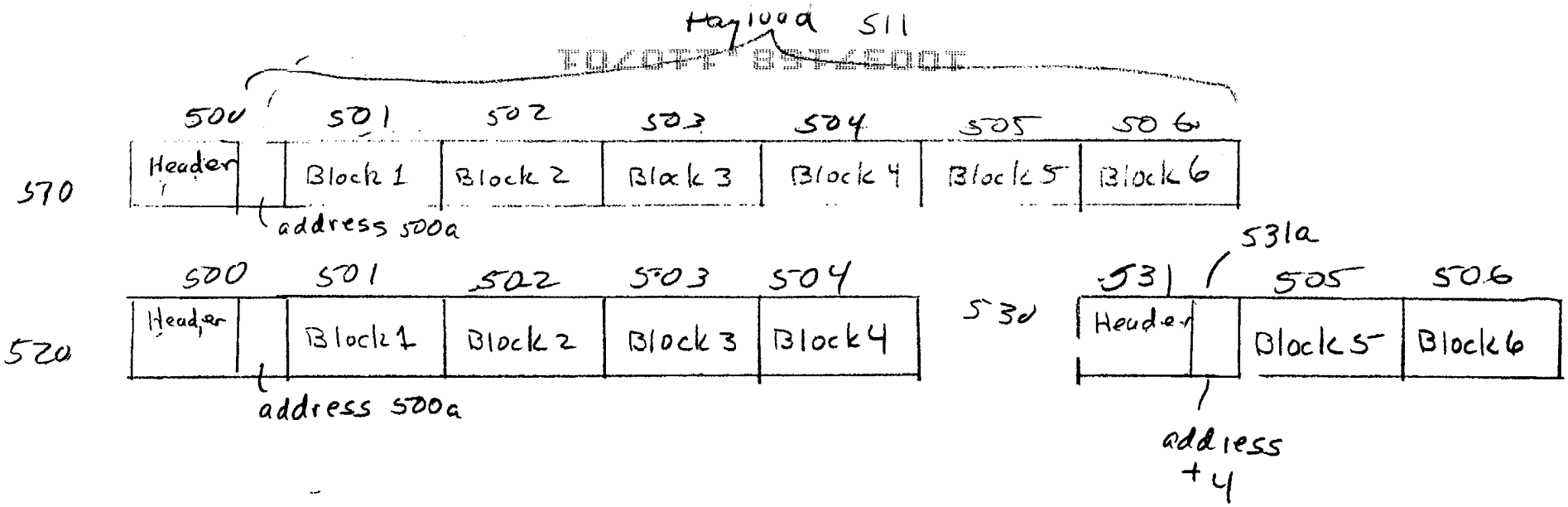

Payload 511
500
501
502
503
504
505
506
570
Header
Block 1
Block 2
Block 3
Block 4
Block 5
Block 6
address 500a
500
501
502
503
504
520
Header
Block 1
Block 2
Block 3
Block 4
address 500a
531a
531
505
506
530
Header
Block 5
Block 6
address
+4

Fig 5

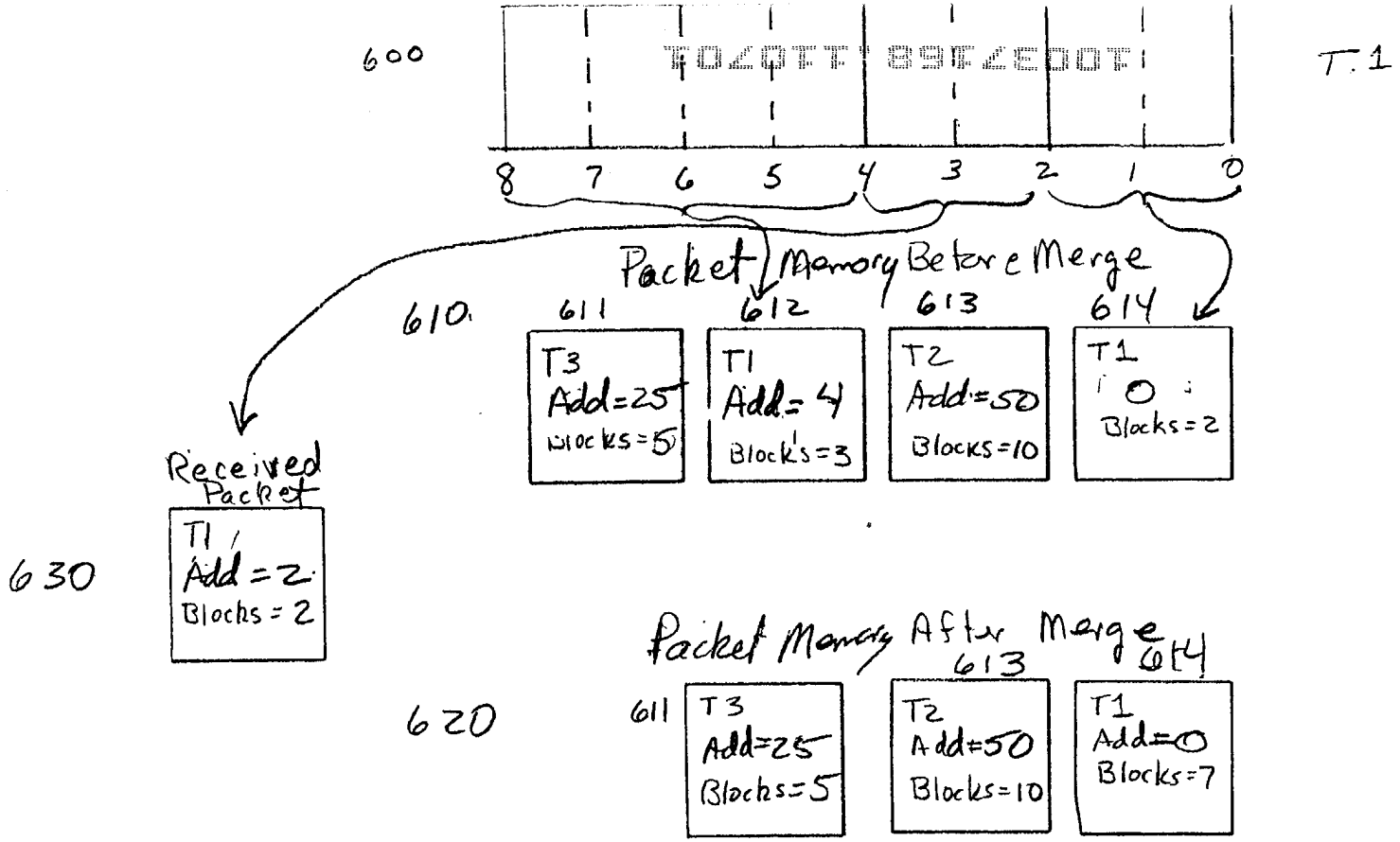

600
T.1
8 7 6 5 4 3 2 1 0
Packet Memory Before Merge
610
611
612
613
614
T3
Add=25
Blocks=5
T1
Add=4
Blocks=3
T2
Add=50
Blocks=10
T1
Add=0
Blocks=2
Received Packet
630
T1
Add=2
Blocks=2
Packet Memory After Merge
620
611
613
614
T3
Add=25
Blocks=5
T2
Add=50
Blocks=10
T1
Add=0
Blocks=7

Fig 6

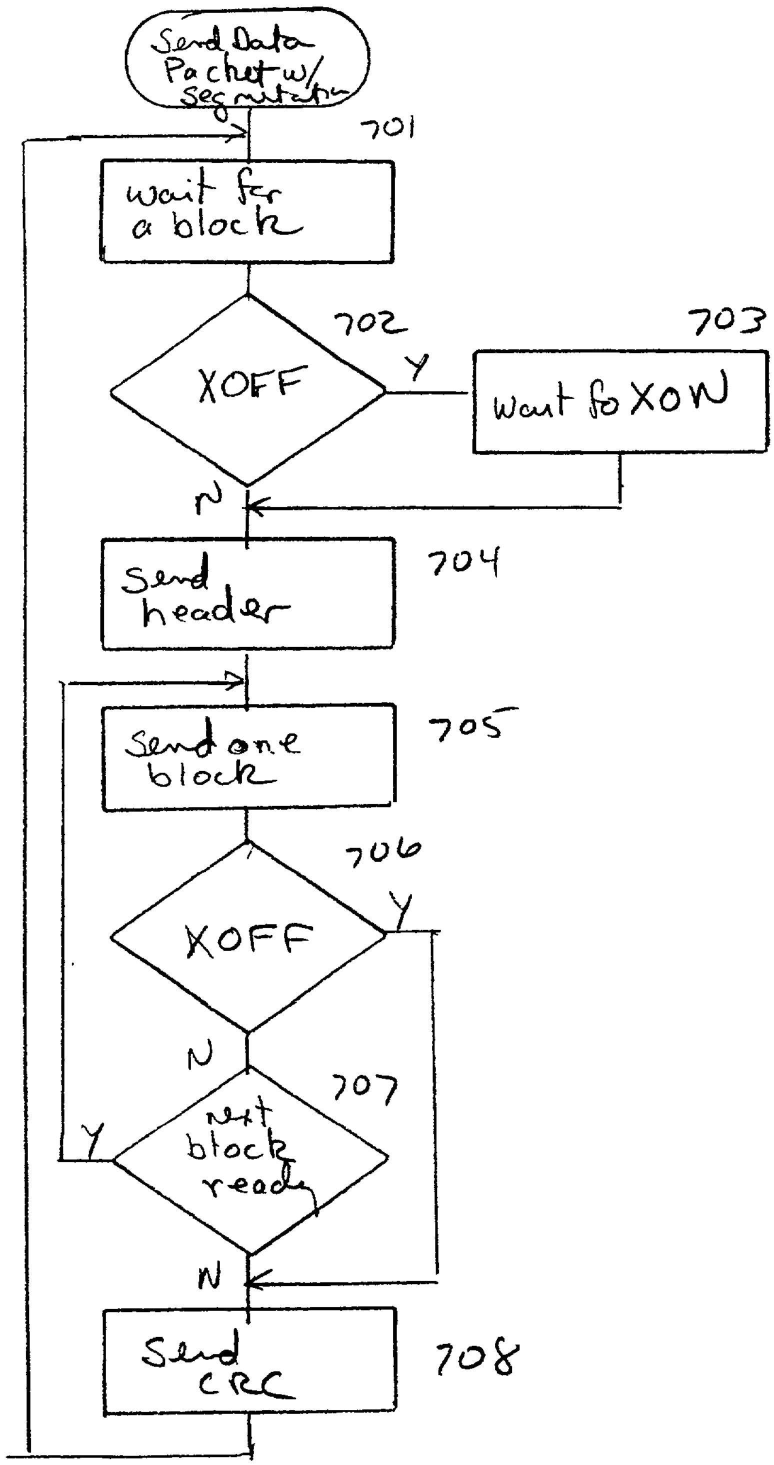


Fig 7

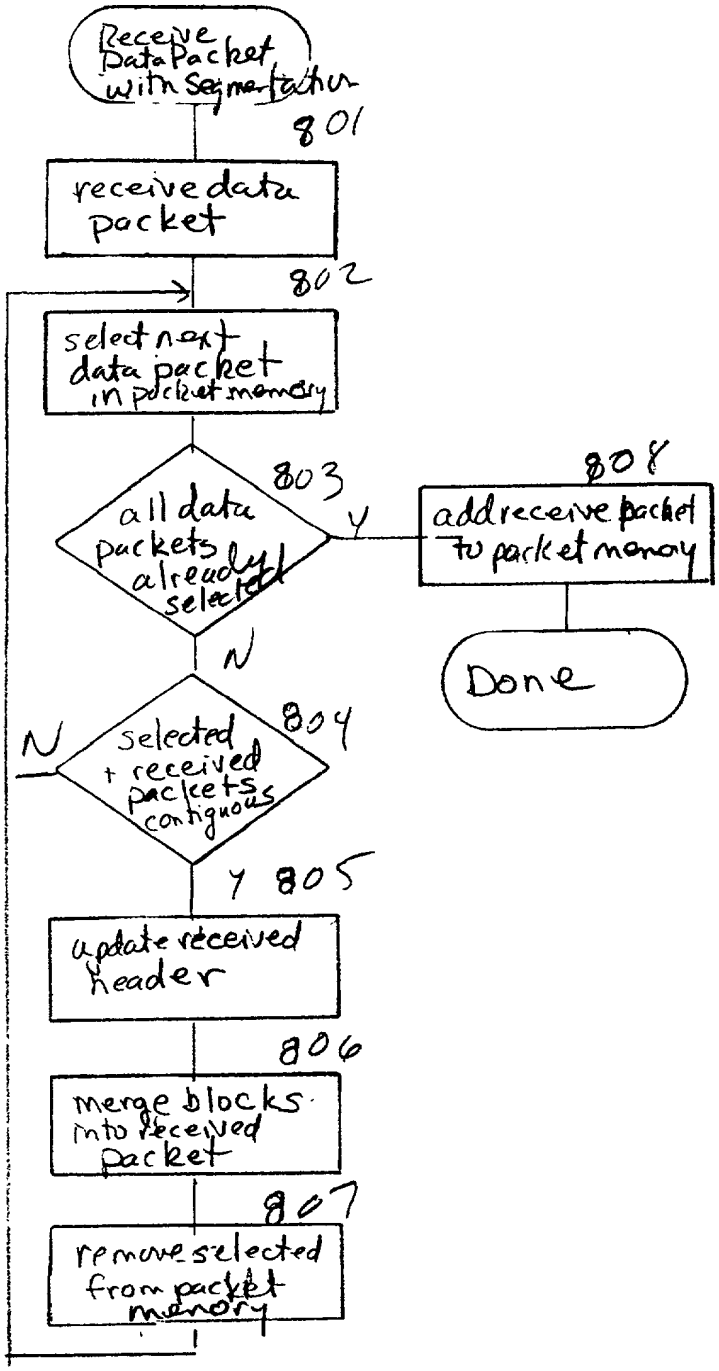
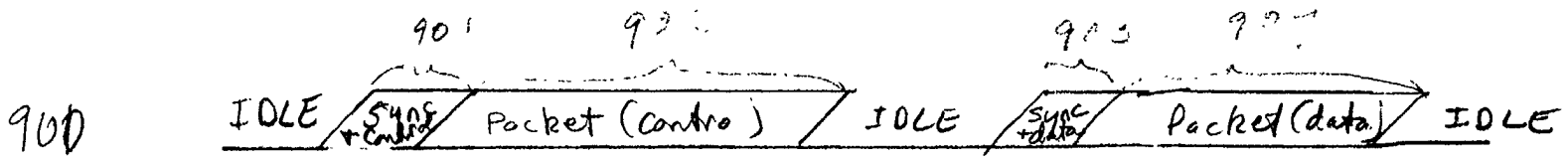

Receive Data Packet with Segmentation
801
receive data packet
802
select next data packet in packet memory
803
all data packets already selected
Y
808
add receive packet to packet memory
Done
N
804
selected + received packets contiguous
N
Y 805
update received header
806
merge blocks into received packet
807
remove selected from packet memory

Fig. 8



Sync + packet type

Fig 9A

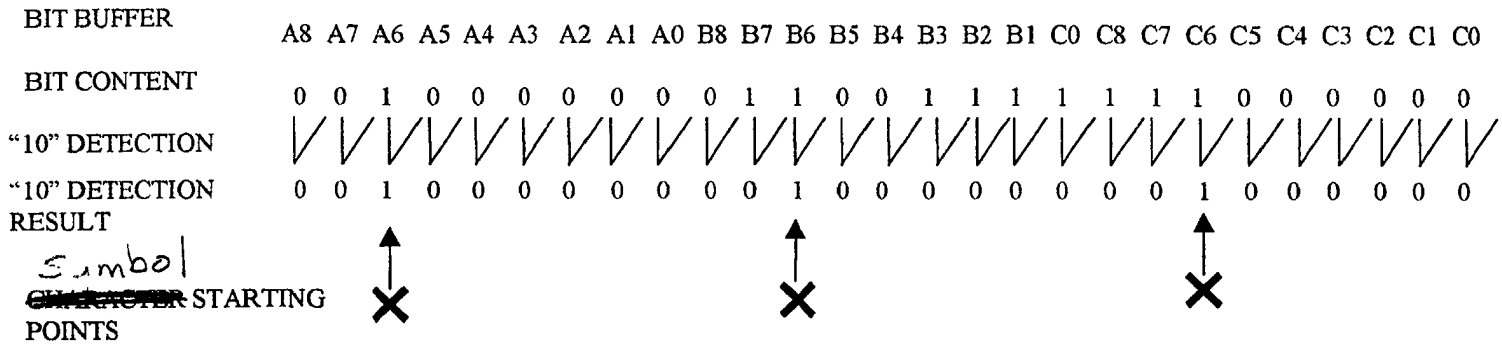


FIG.10

Fig 9B

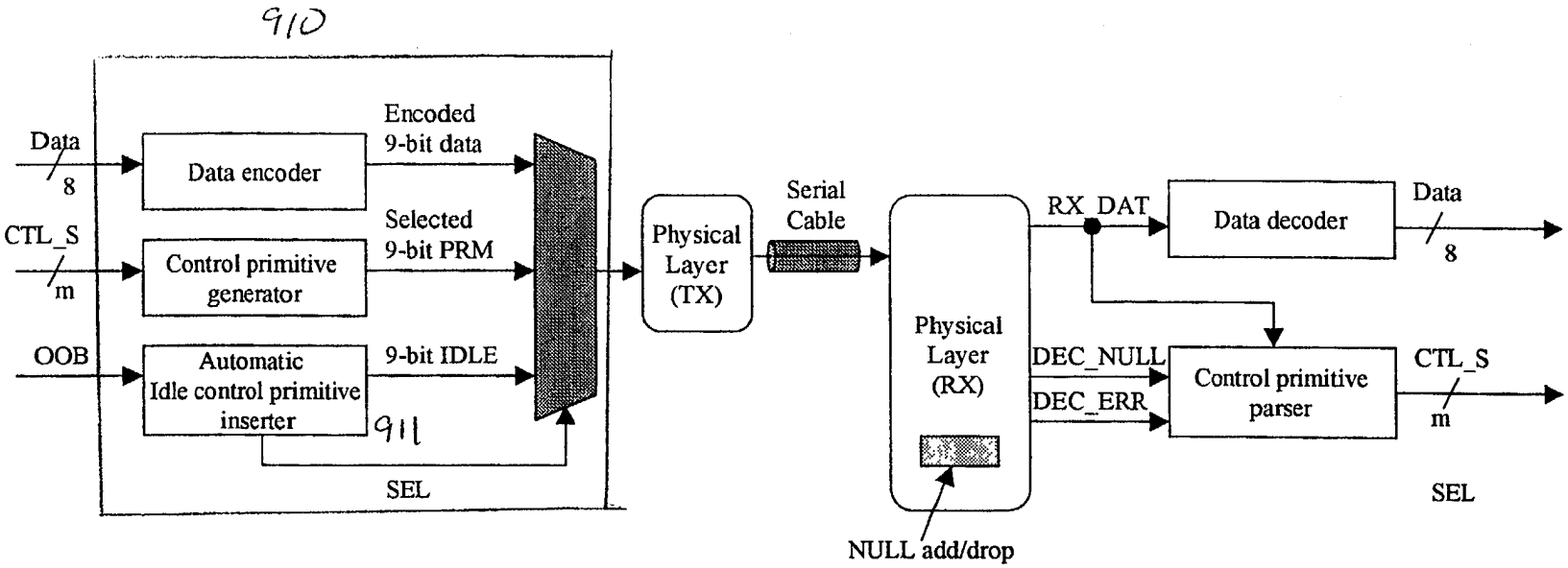


Fig. 9C

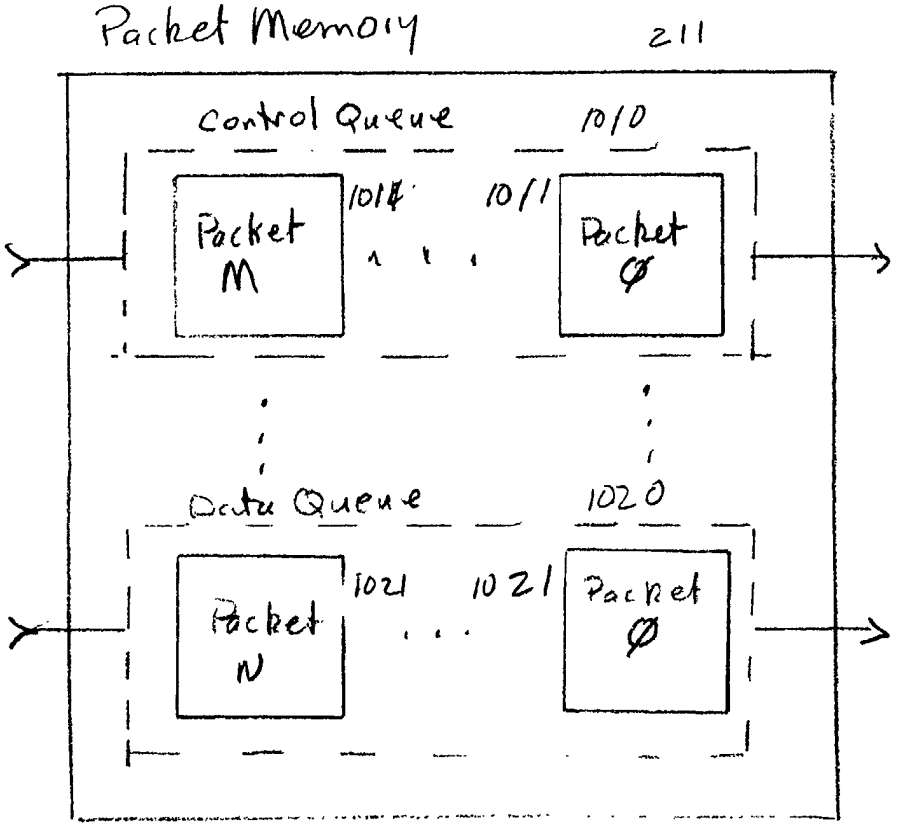


Fig 10

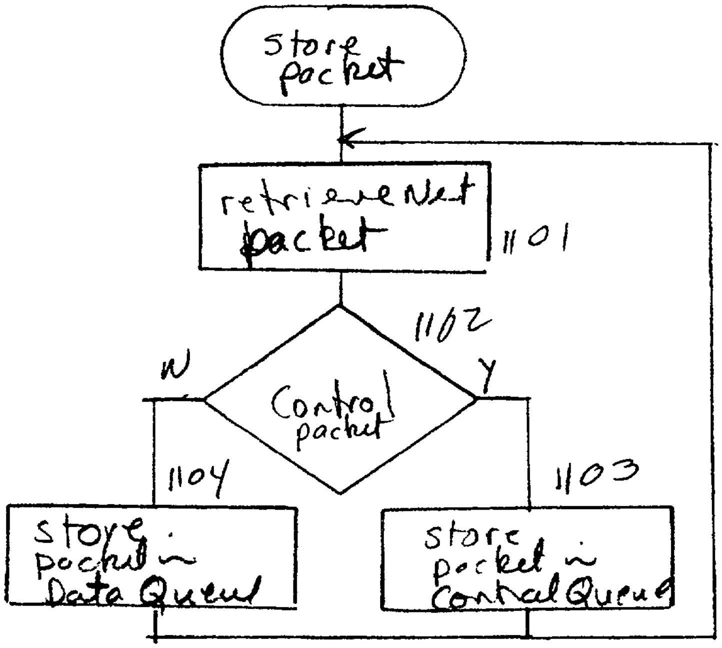


Fig 11

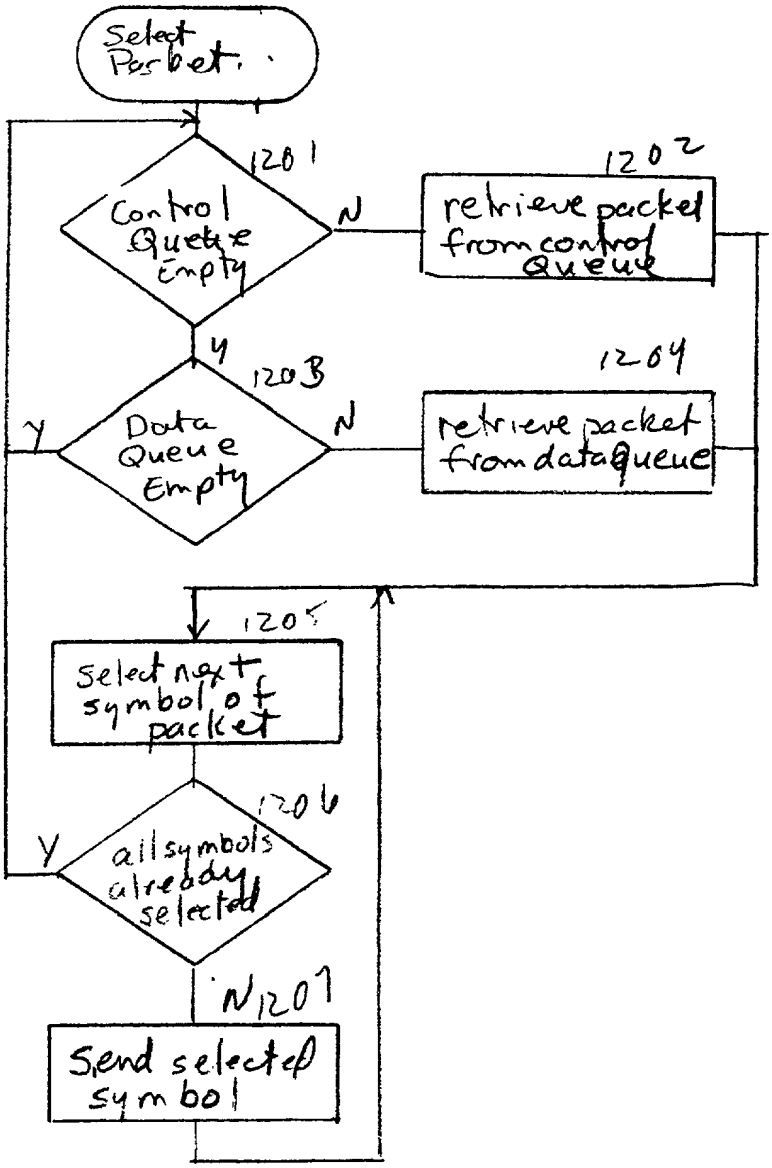


Fig 12

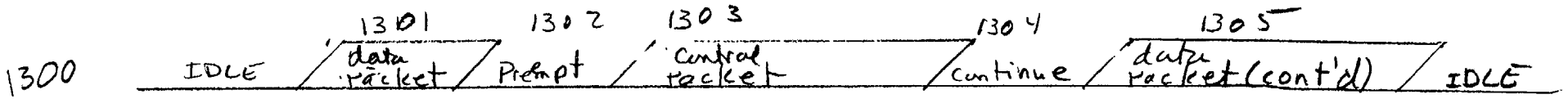


Fig 13

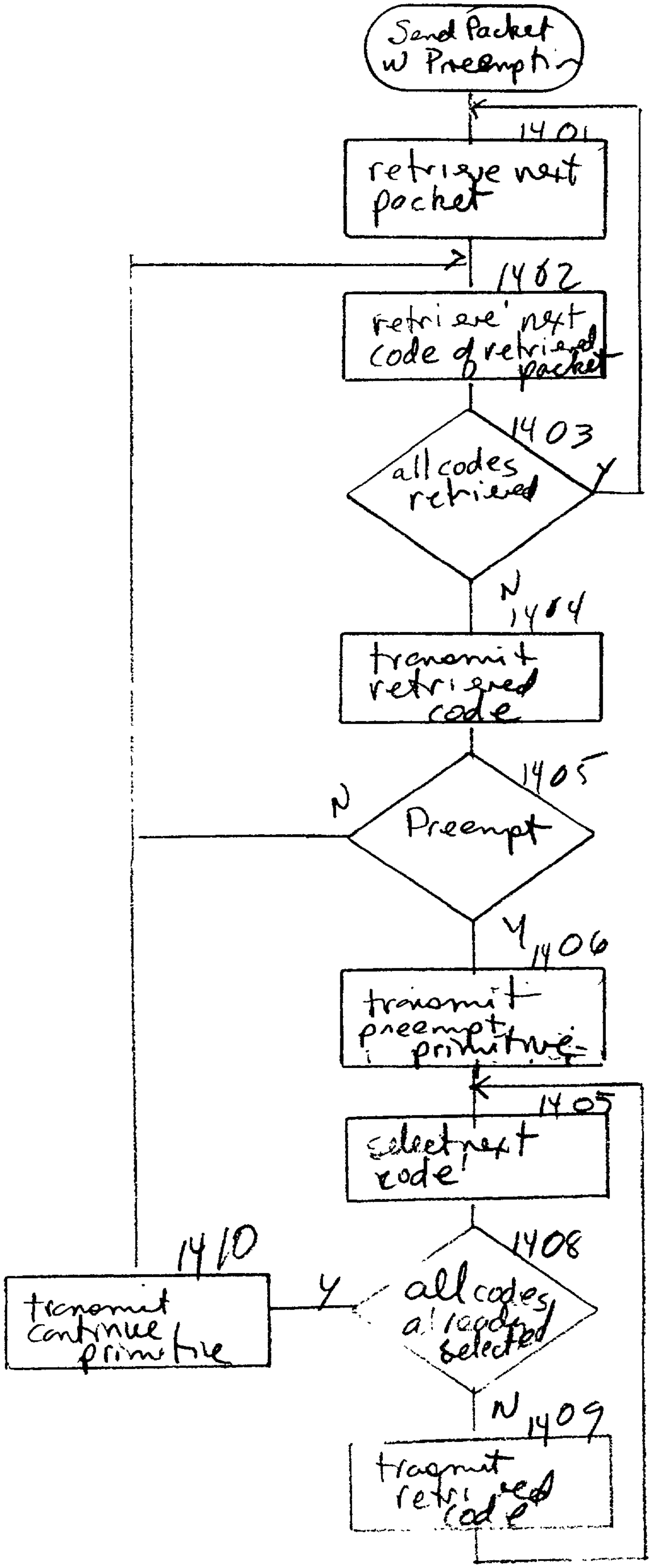

Send Packet w Preemption
1401
retrieve next packet
1402
retrieve next code of retrieved packet
1403
all codes retrieved
Y
N
1404
transmit retrieved code
1405
Preempt
N
Y
1406
transmit preempt primitive
1405
select next code
1408
all codes already selected
Y
1410
transmit continue primitive
N
1409
transmit retrieved code

Fig14

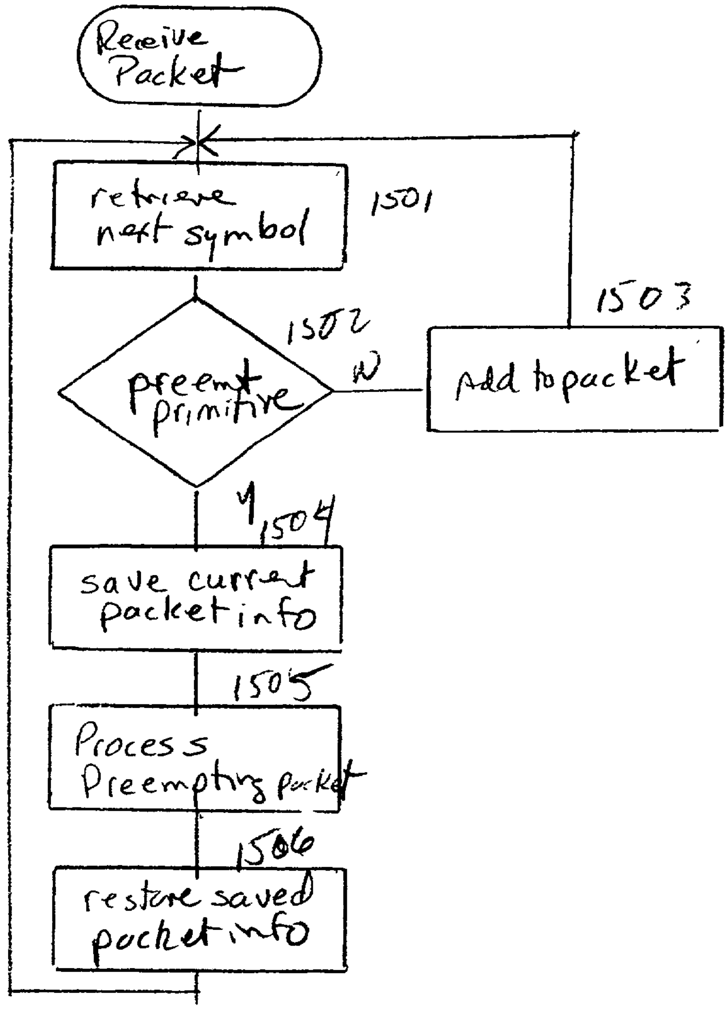


Fig 15

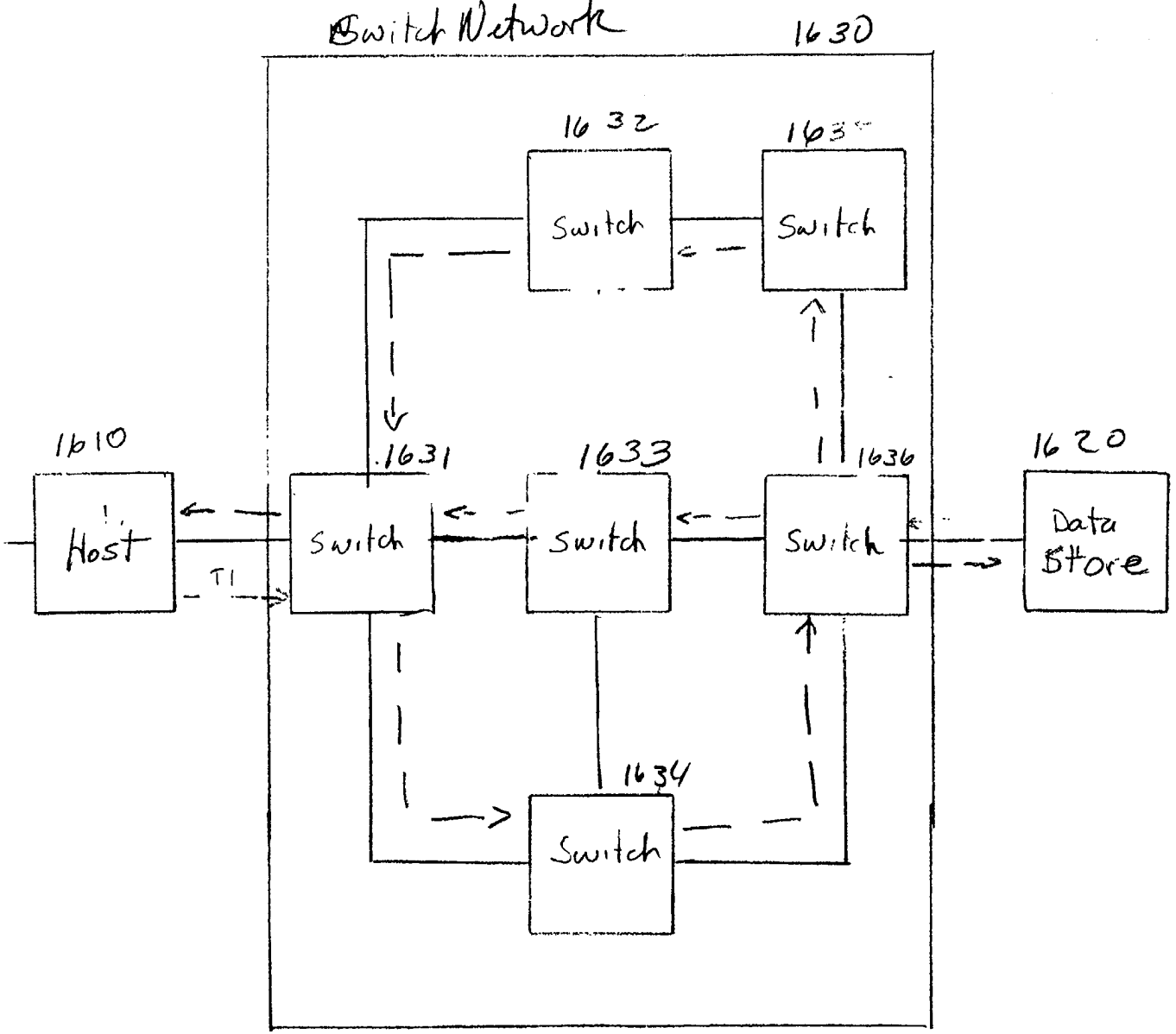


Fig 16

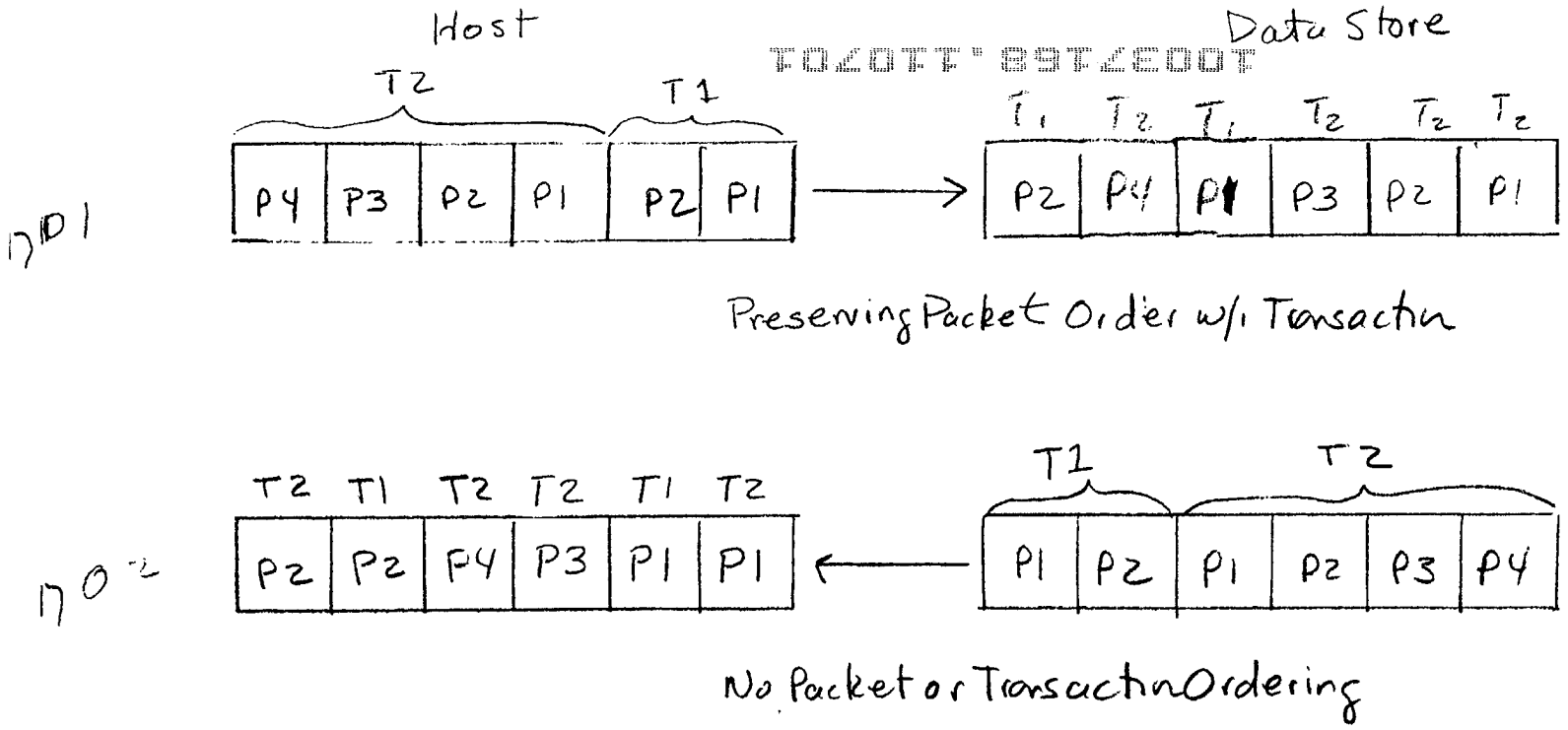

Host
Data Store
T2
T1
P4
P3
P2
P1
P2
P1
T1
T2
T1
T2
T2
T2
P2
P4
P1
P3
P2
P1
Preserving Packet Order w/ Transaction
T2
T1
T2
T2
T1
T2
P2
P2
P4
P3
P1
P1
T1
T2
P1
P2
P1
P2
P3
P4
No Packet or Transaction Ordering

Fig 17

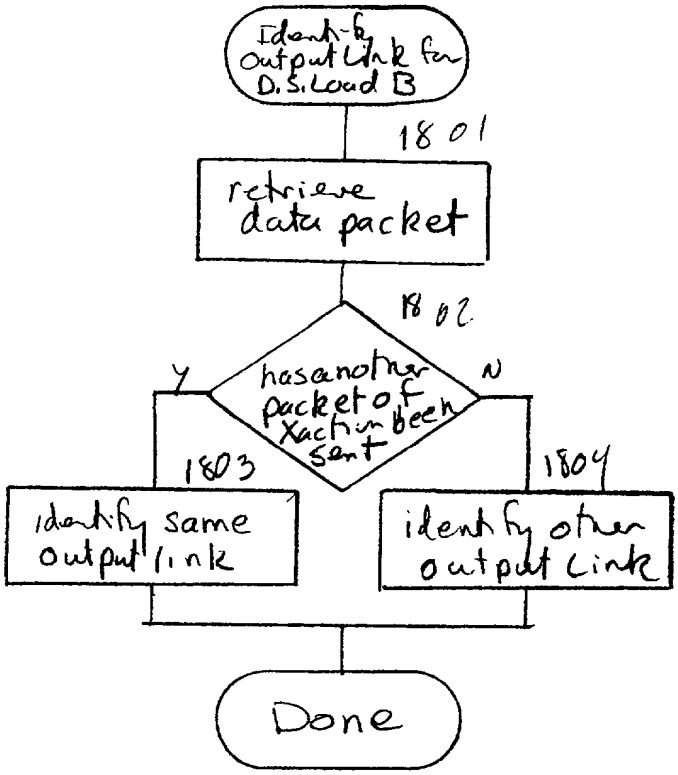


Fig 18

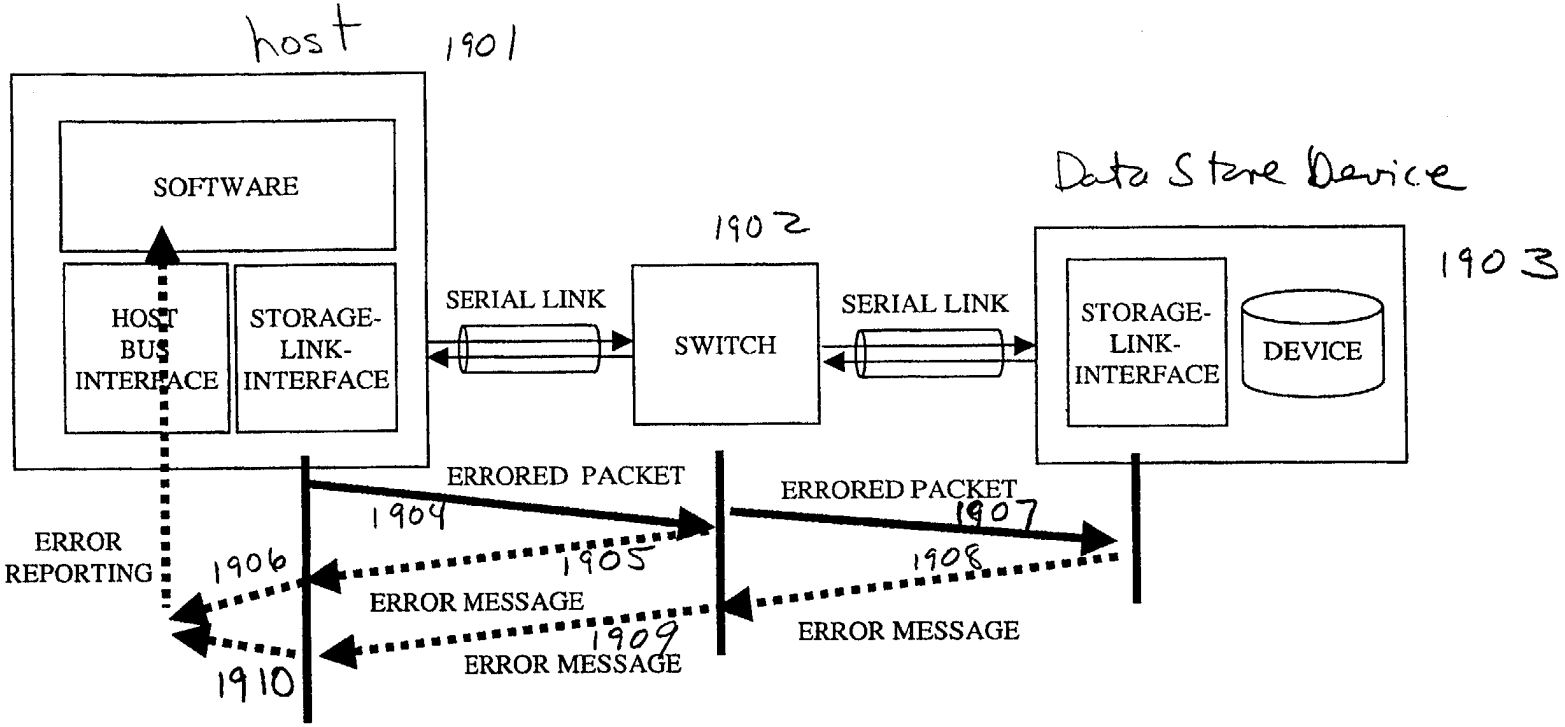

host
1901
SOFTWARE
HOST BUS INTERFACE
STORAGE-LINK-INTERFACE
SERIAL LINK
1902
SWITCH
SERIAL LINK
Data Store Device
1903
STORAGE-LINK-INTERFACE
DEVICE
ERRORED PACKET
1904
ERRORED PACKET
1907
ERROR REPORTING
1906
1905
1908
ERROR MESSAGE
ERROR MESSAGE
1909
1910
ERROR MESSAGE

Fig 19A

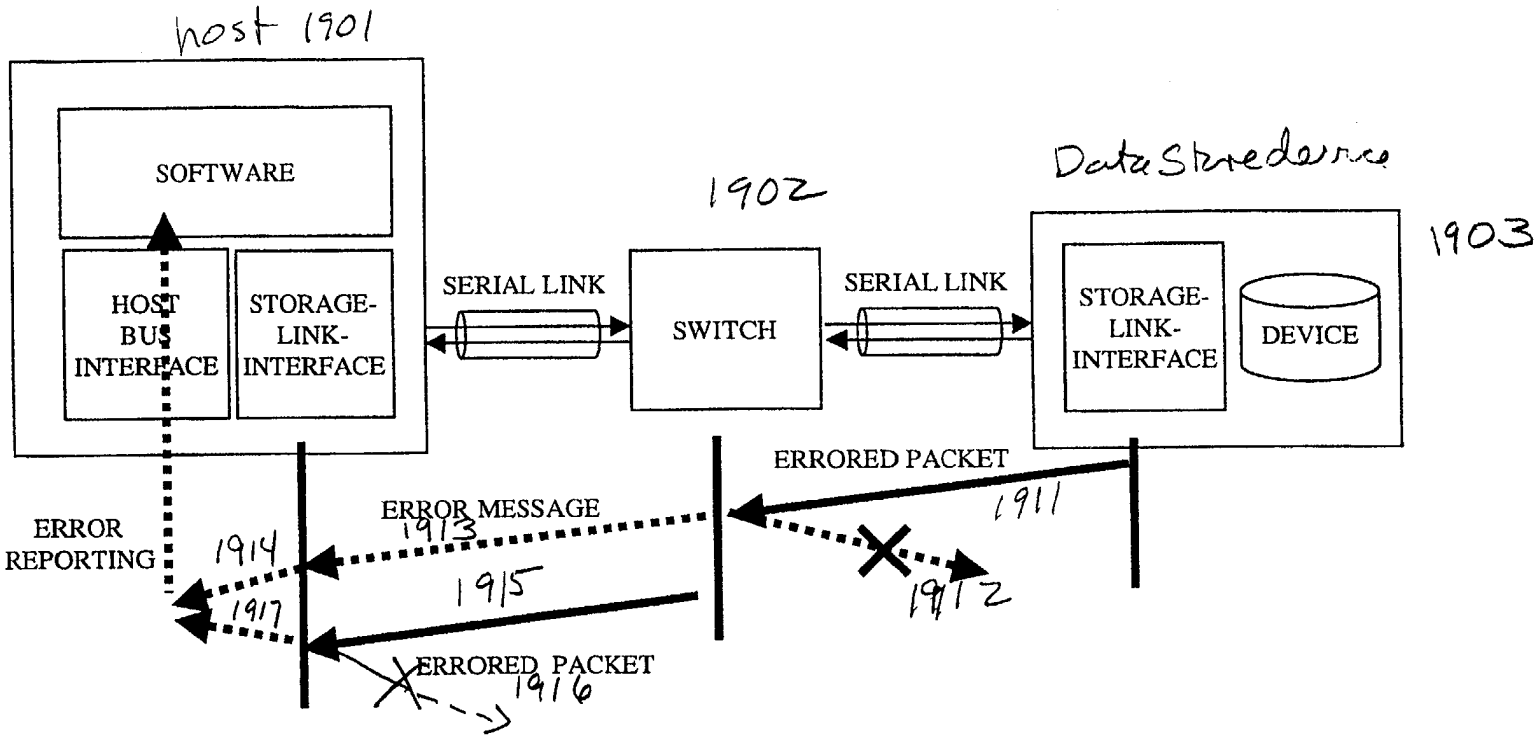
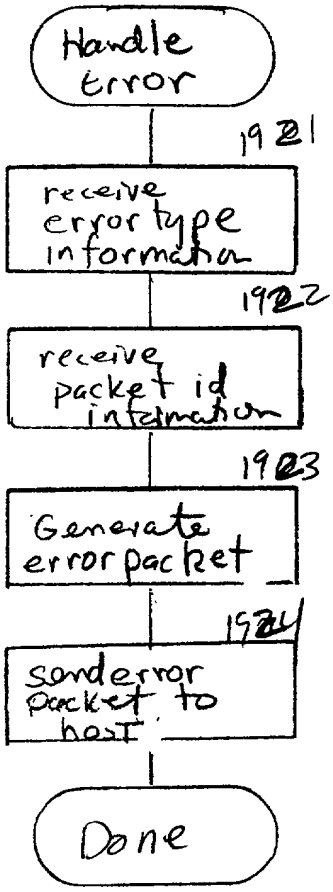


Fig 19B


Handle error
1921
receive error type information
1922
receive packet id information
1923
Generate error packet
1924
send error packet to host
Done

19C

8b Code	9 bit symbol
0000 0000	101010101
0000 0001	101010100
0000 0010	101010111
⋮	
0101 0101	001010101
⋮	
0111 0110	001110110
0111 0111	100100010
⋮	
1111 1111	110101010

Fig 20

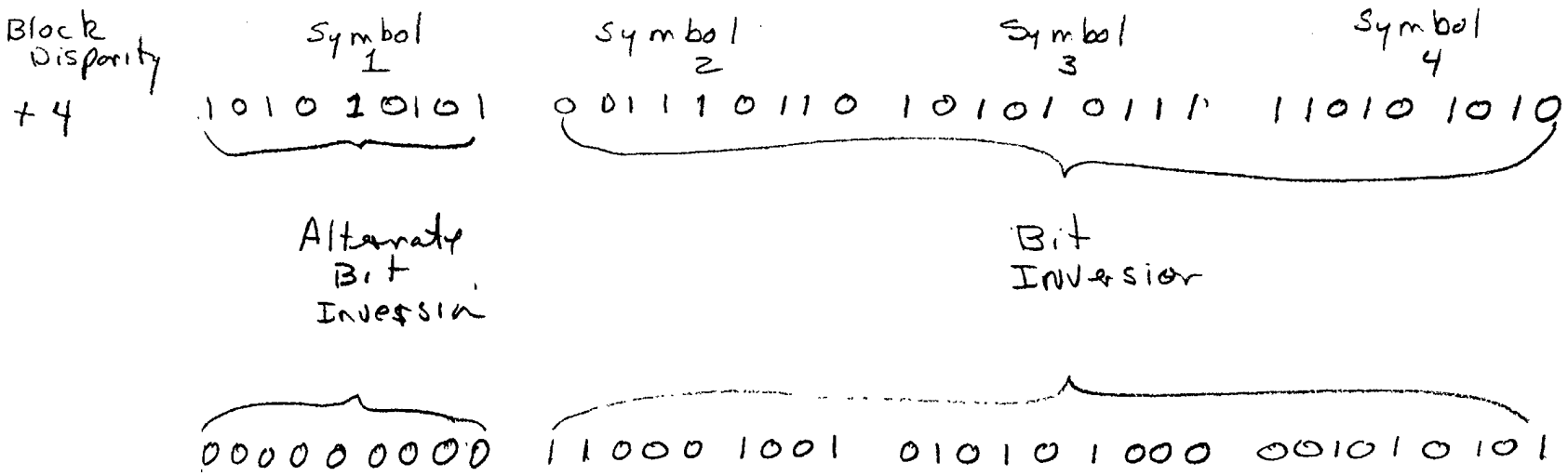


Fig 21A

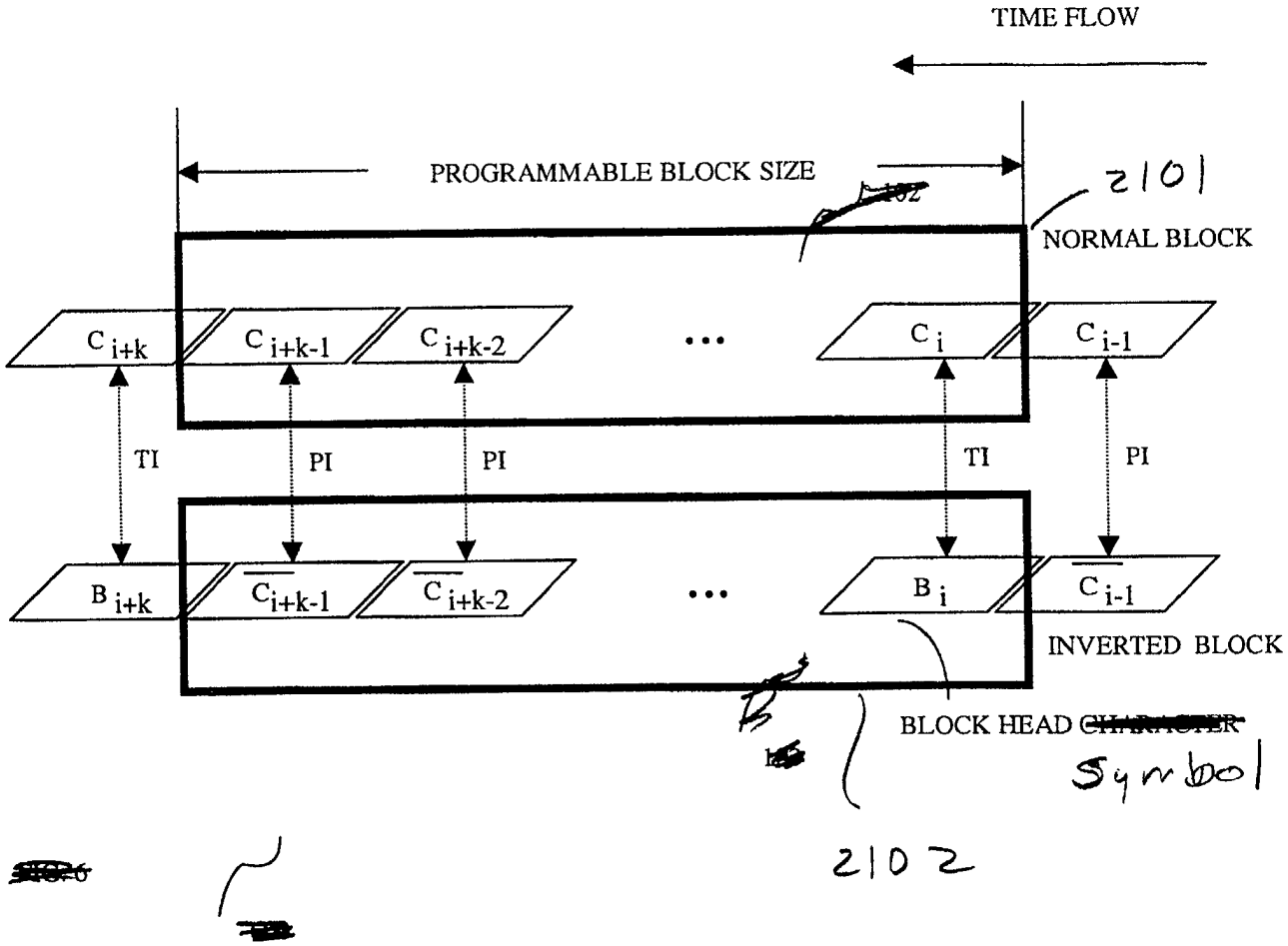


Fig 21B

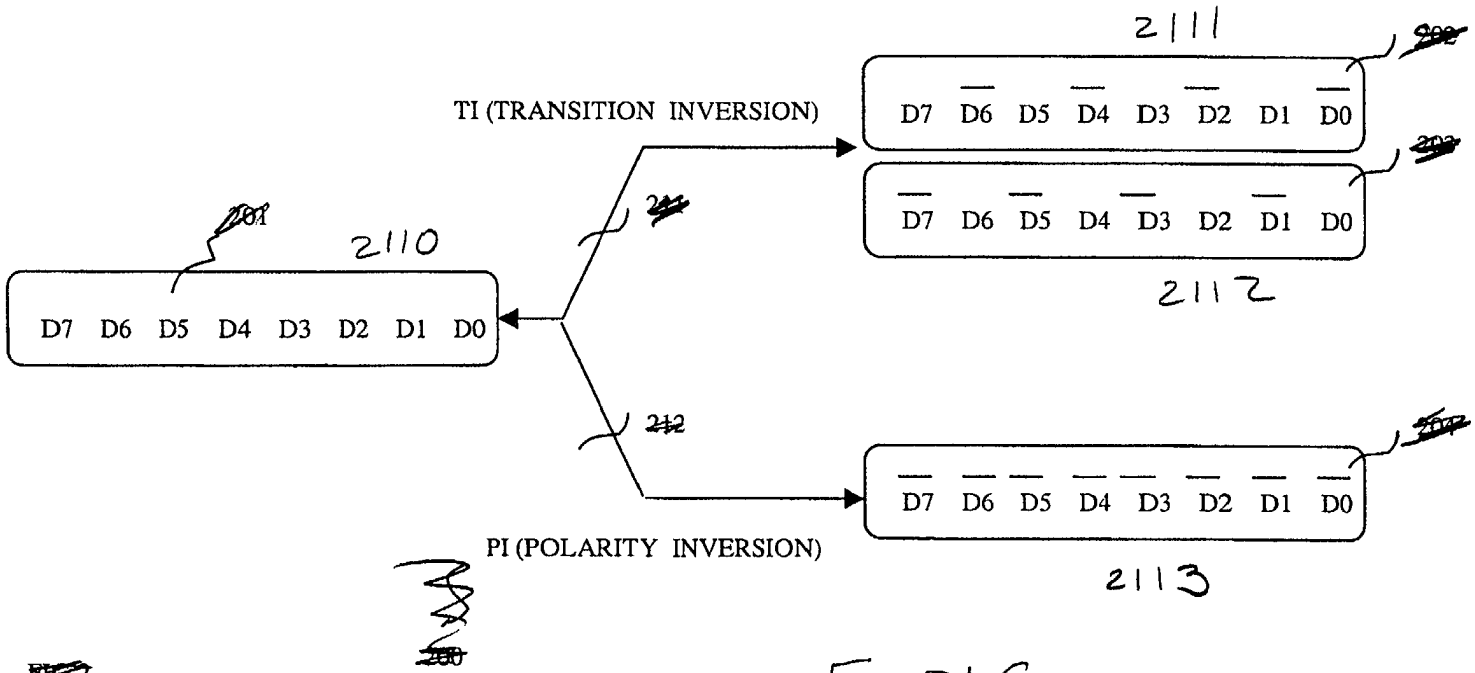

2111
TI (TRANSITION INVERSION)
D7 D6 D5 D4 D3 D2 D1 D0
2110
D7 D6 D5 D4 D3 D2 D1 D0
2112
D7 D6 D5 D4 D3 D2 D1 D0
PI (POLARITY INVERSION)
2113
D7 D6 D5 D4 D3 D2 D1 D0

FIG.2

Fig 21C

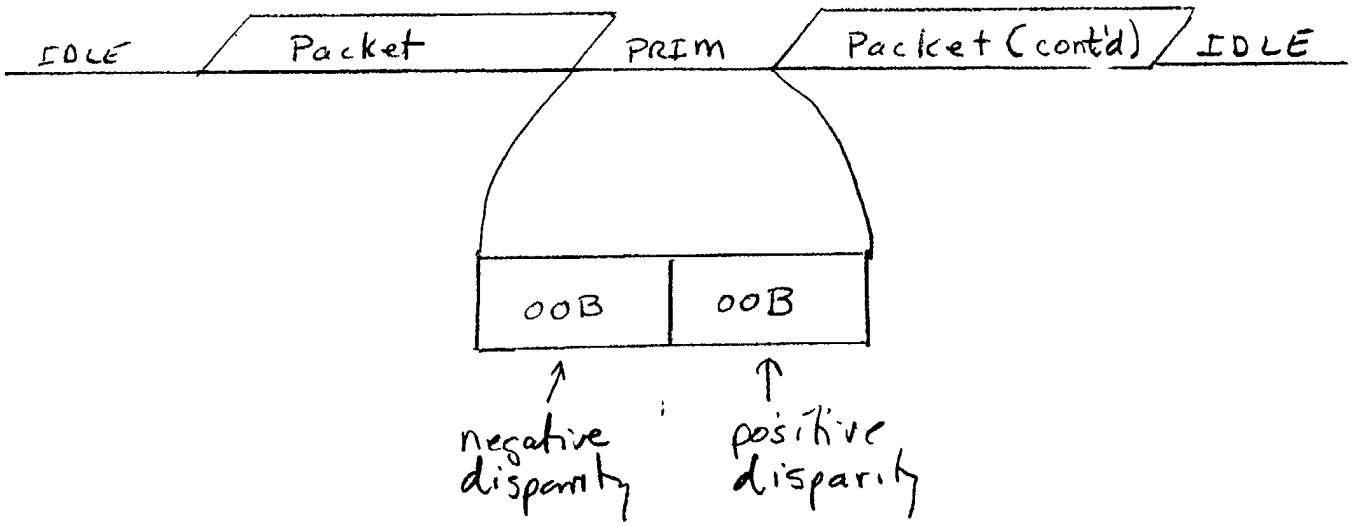

IDLE
Packet
PRIM
Packet (cont'd)
IDLE
OOB
OOB
negative disparity
positive disparity

Fig 22

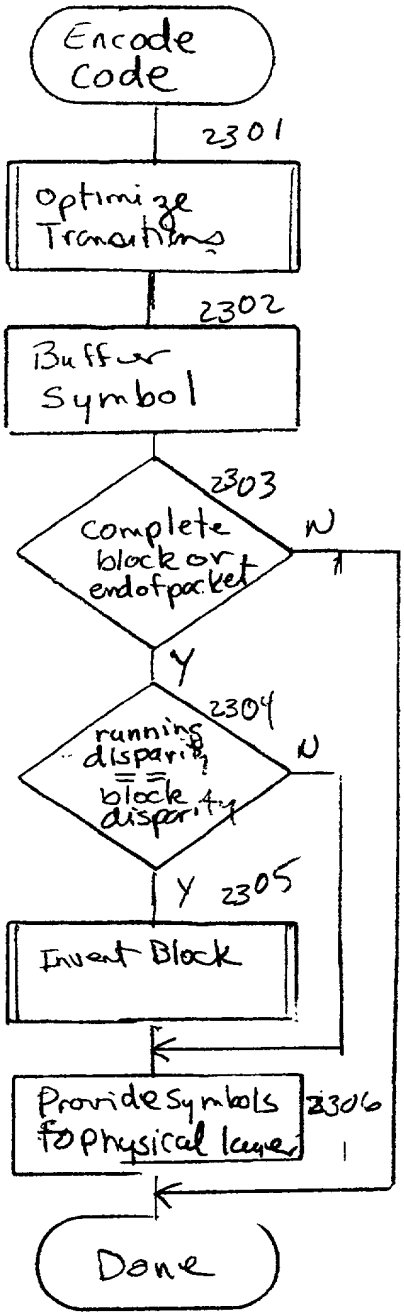

Encode Code
2301
Optimize Transitions
2302
Buffer Symbol
2303
complete block or end of packet
N
Y
2304
running disparity == block disparity
N
Y
2305
Invert Block
Provide Symbols to Physical Layer
2306
Done

Fig 23

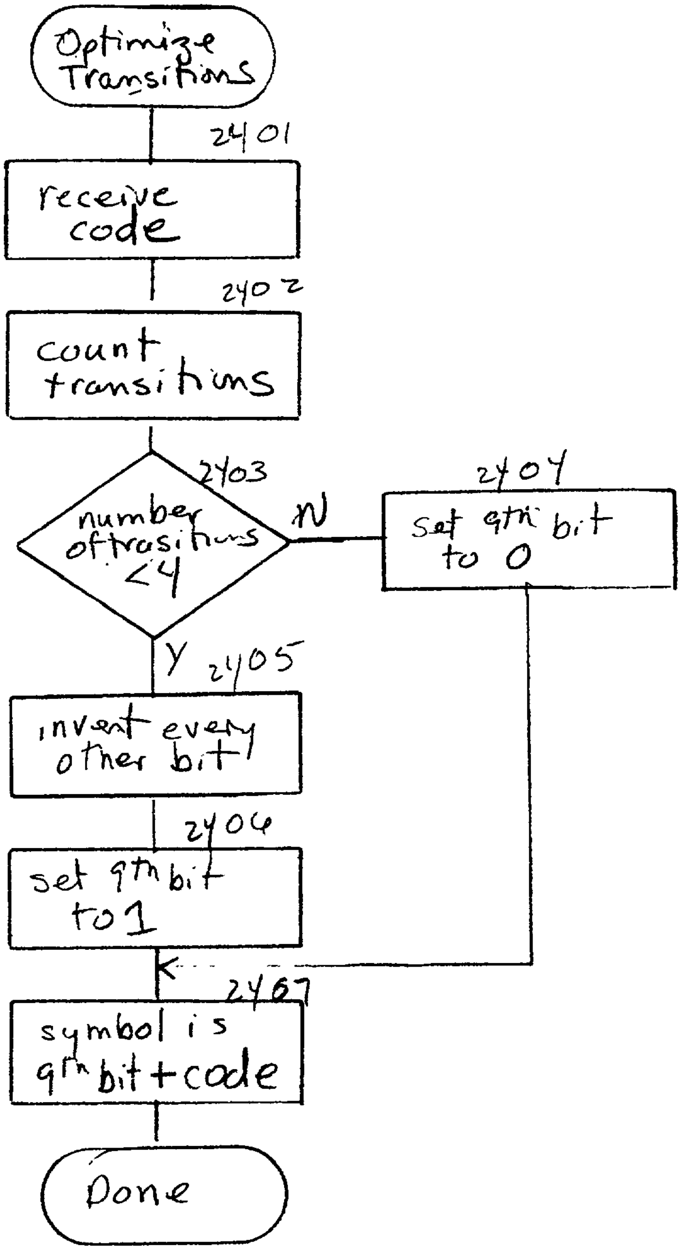


Fig 24

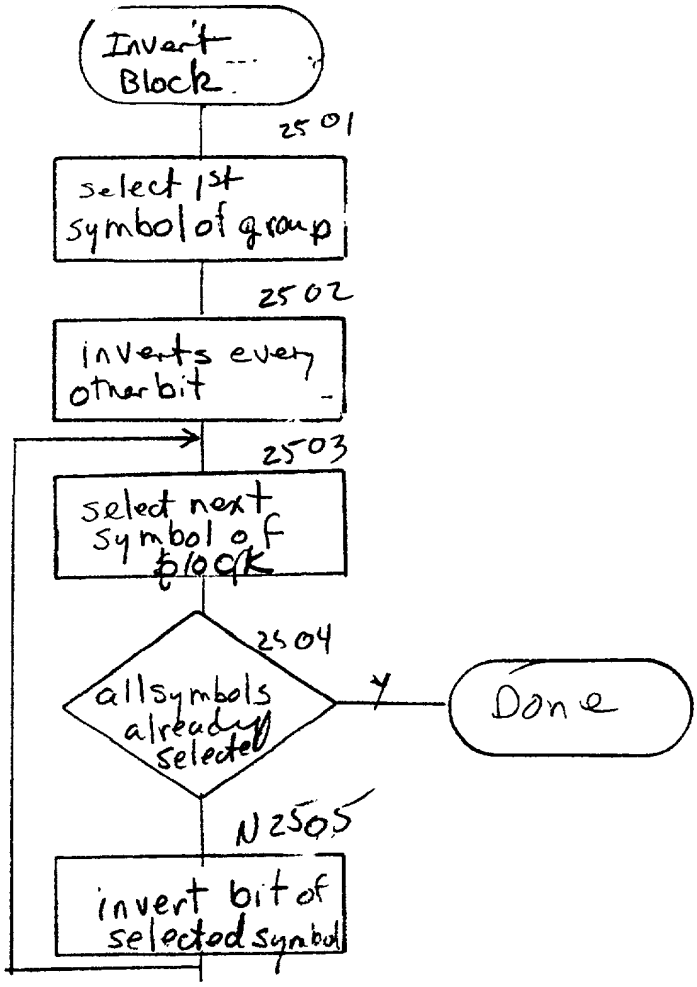

Invert Block
2501
select 1st symbol of group
2502
inverts every other bit
2503
select next symbol of block
2504
all symbols already selected
Y
Done
N
2505
invert bit of selected symbol

Fig 25

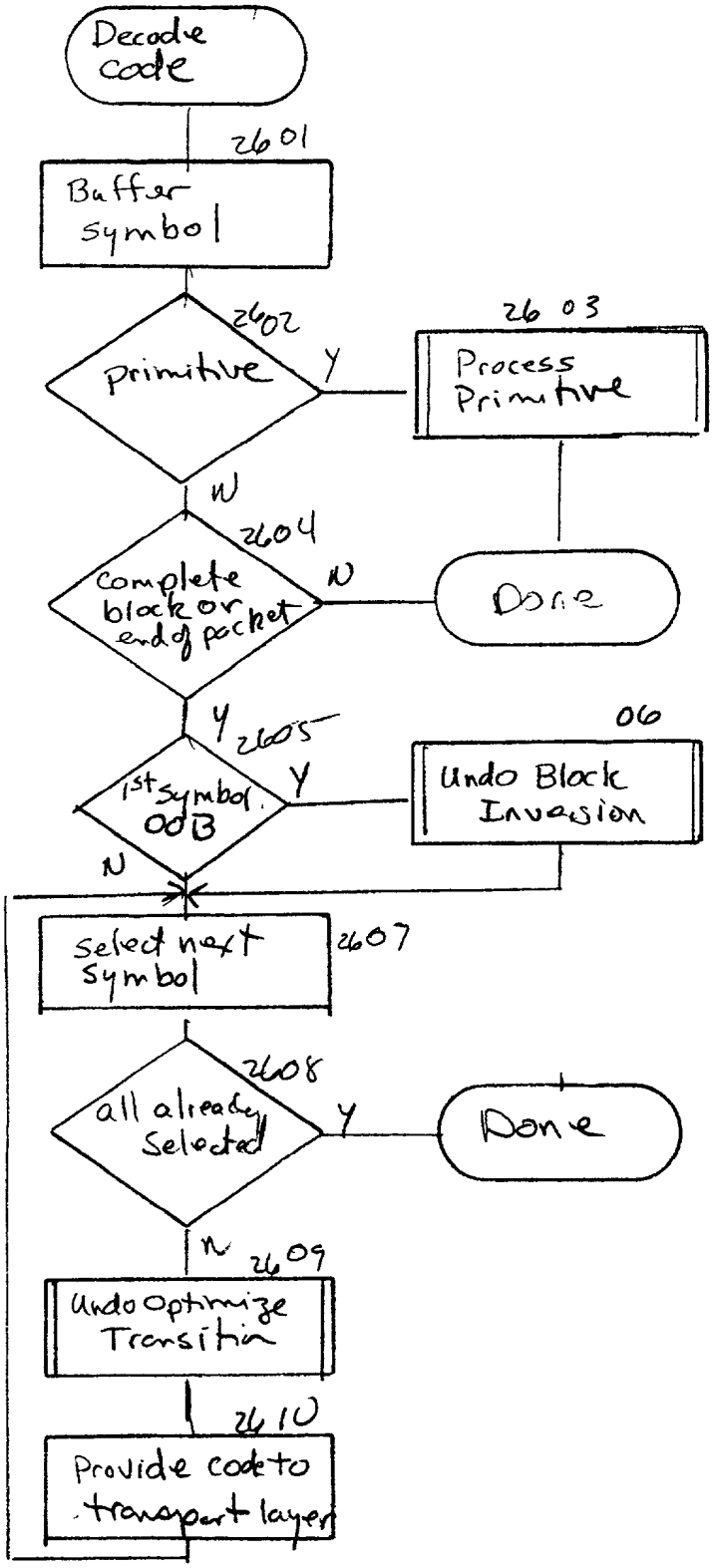


Fig 26

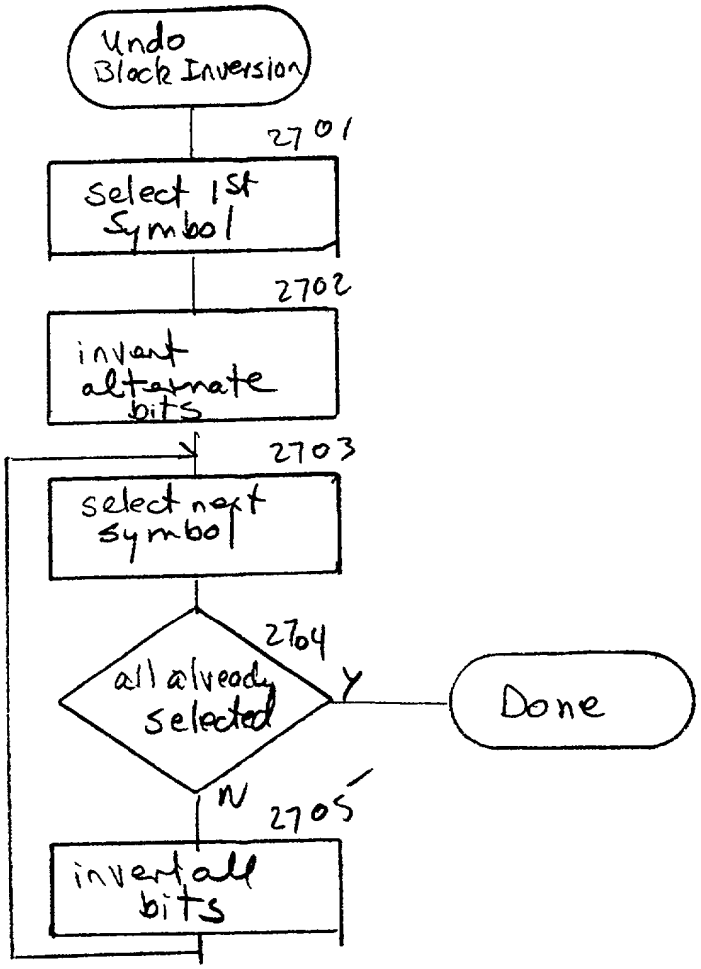

Undo Block Inversion
2701
Select 1st Symbol
2702
invert alternate bits
2703
select next symbol
2704
all already selected
Y
Done
N
2705
invert all bits

Fig 27

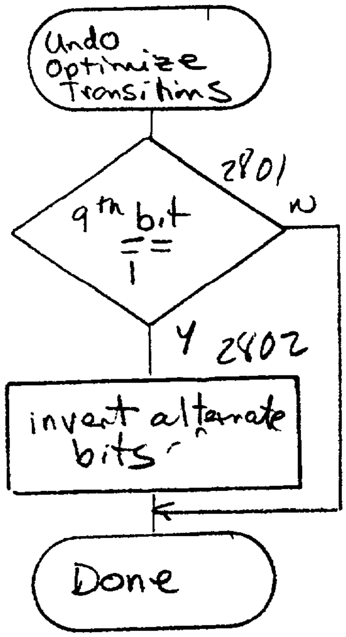

Undo Optimize Transitions
2801
N
9th bit == 1
Y
2802
invert alternate bits
Done

Fig 28

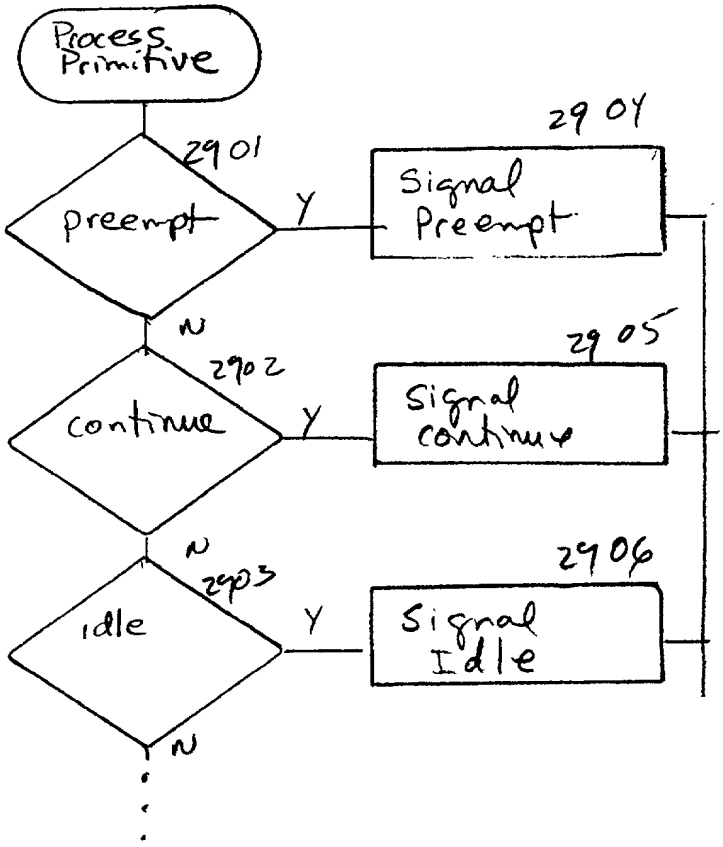


Fig 29

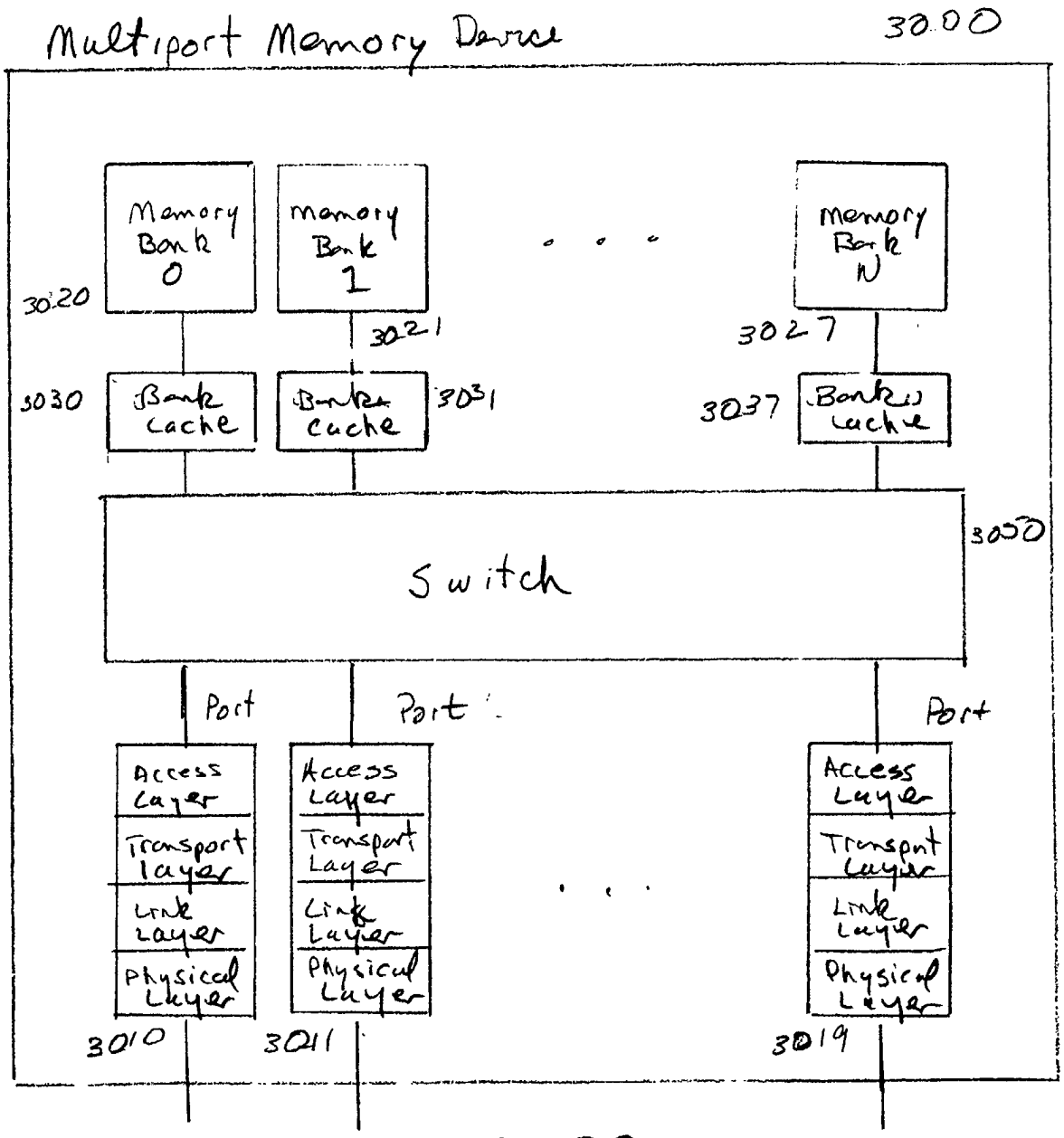


Fig 30

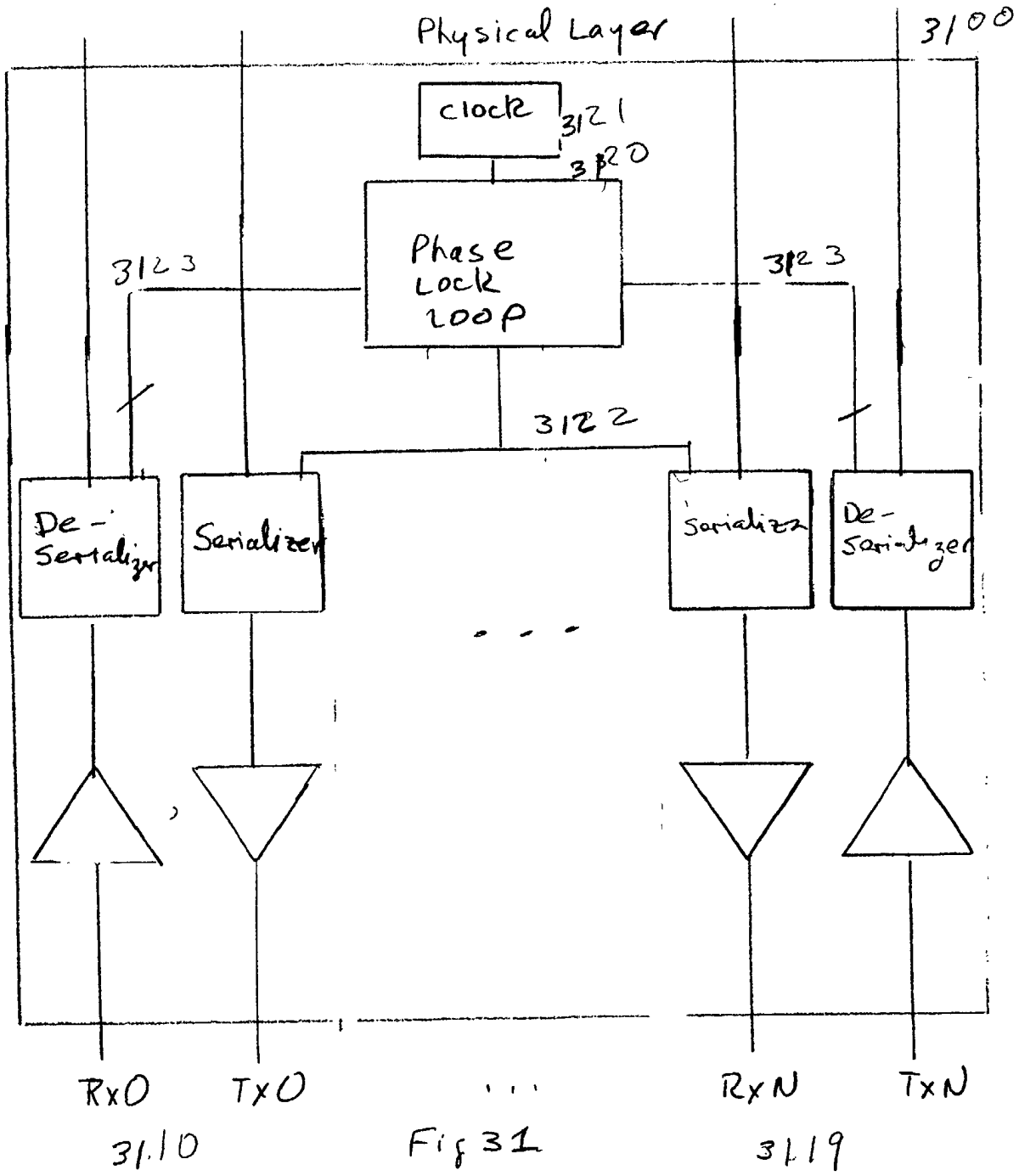


Fig 31

Input Queue 3201

Port	R/W	Address	Data
3	R	1000	
4	W	4000	10....1
3	W	1000	111...0
3	R	2000	
	⋮		

Output Queue 3202

Valid	Port	Data
1	3	11...0
0		
0		
1	3	101...1
	⋮	

Fig 32

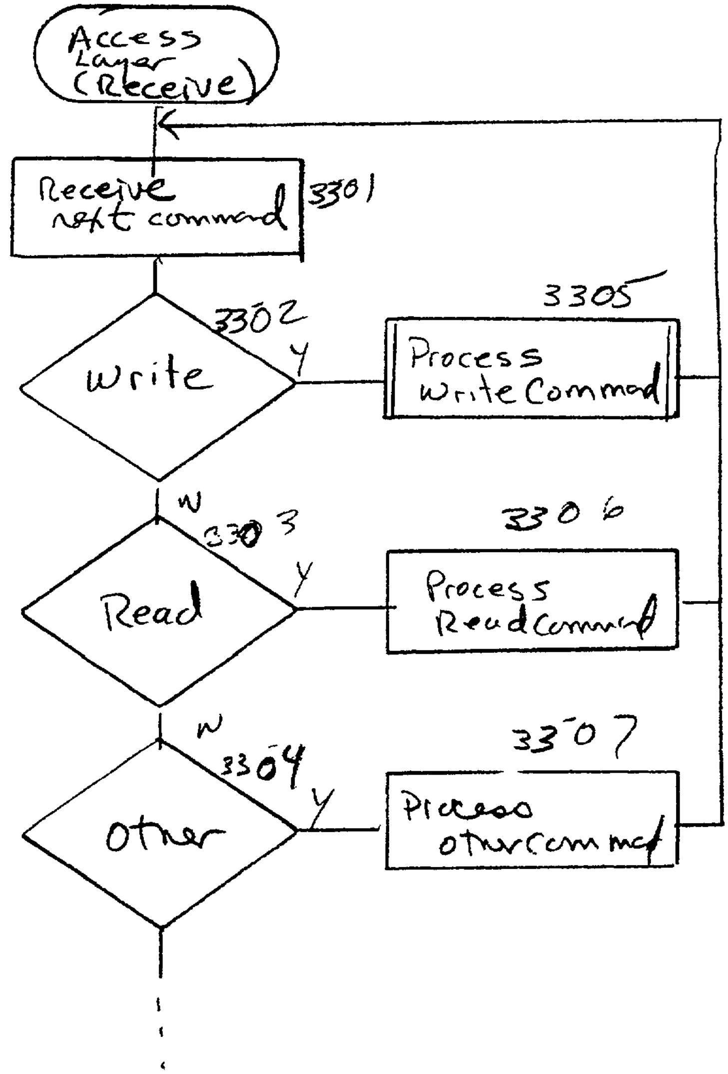

Access Layer (Receive)
Receive next command
3301
Write
3302
Y
N
Process Write Command
3305
Read
3303
Y
N
Process Read Command
3306
Other
3304
Y
Process other command
3307

Fig 33

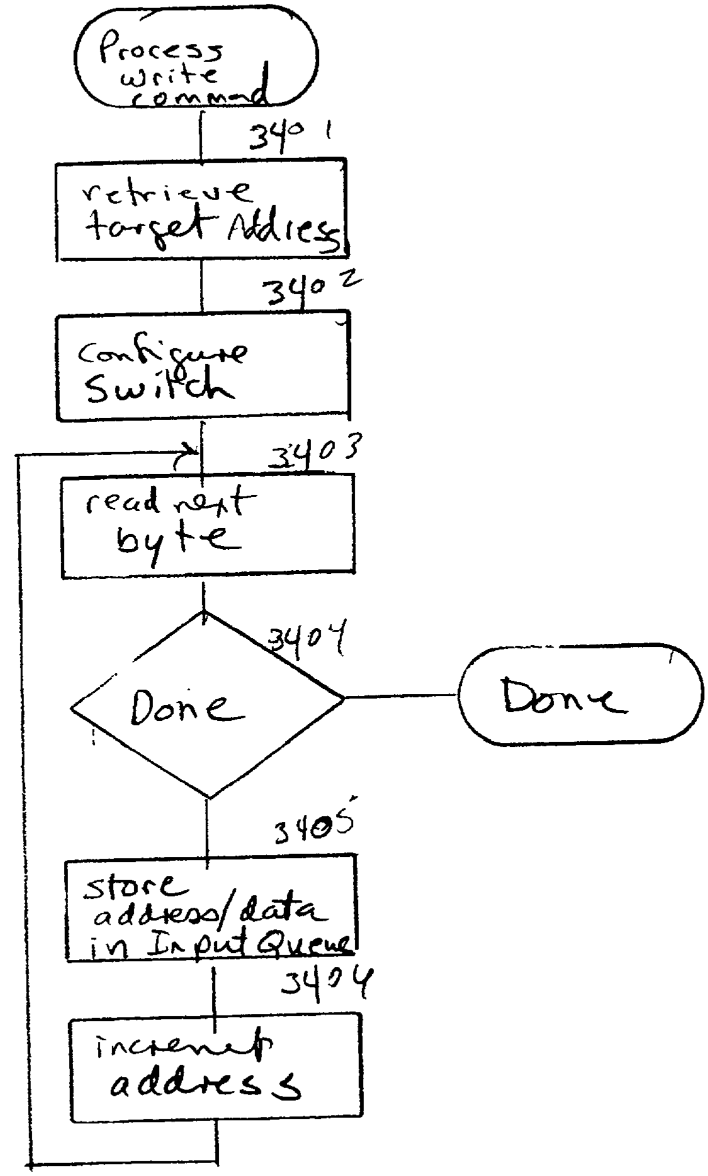

Process write command
3401
retrieve target Address
3402
Configure Switch
3403
read next byte
3404
Done
Done
3405
store address/data in Input Queue
3404
increment address

Fig 34

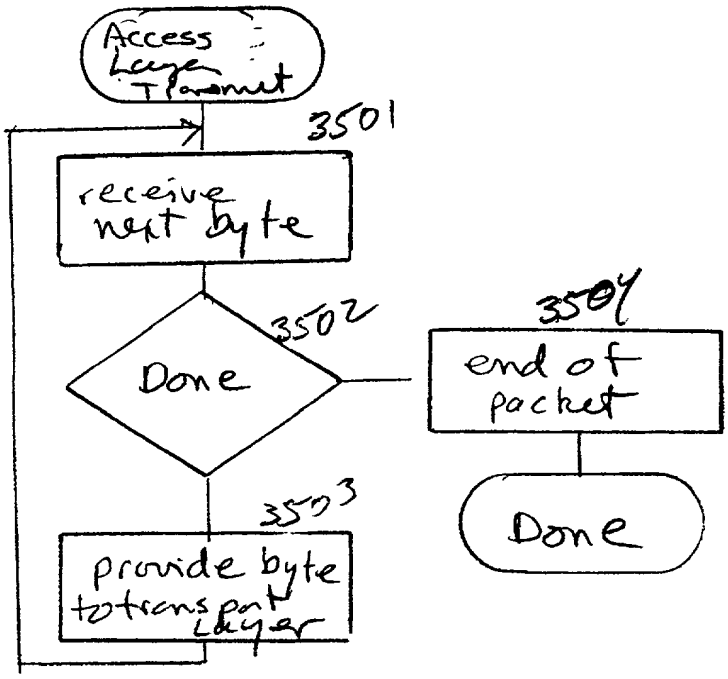

Access Layer Transmit
3501
receive next byte
3502
Done
3504
end of packet
Done
3503
provide byte to transport Layer

Fig 35

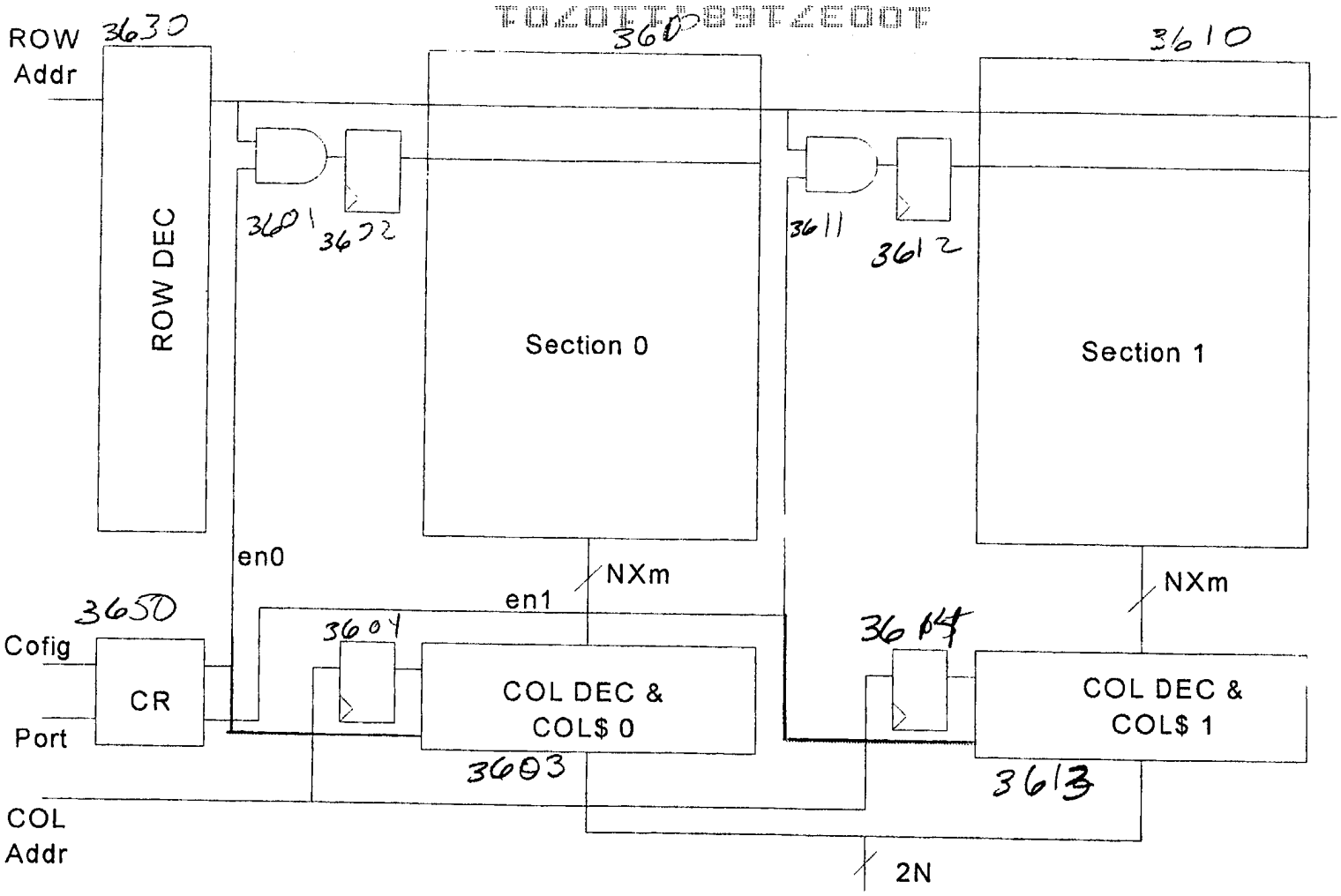
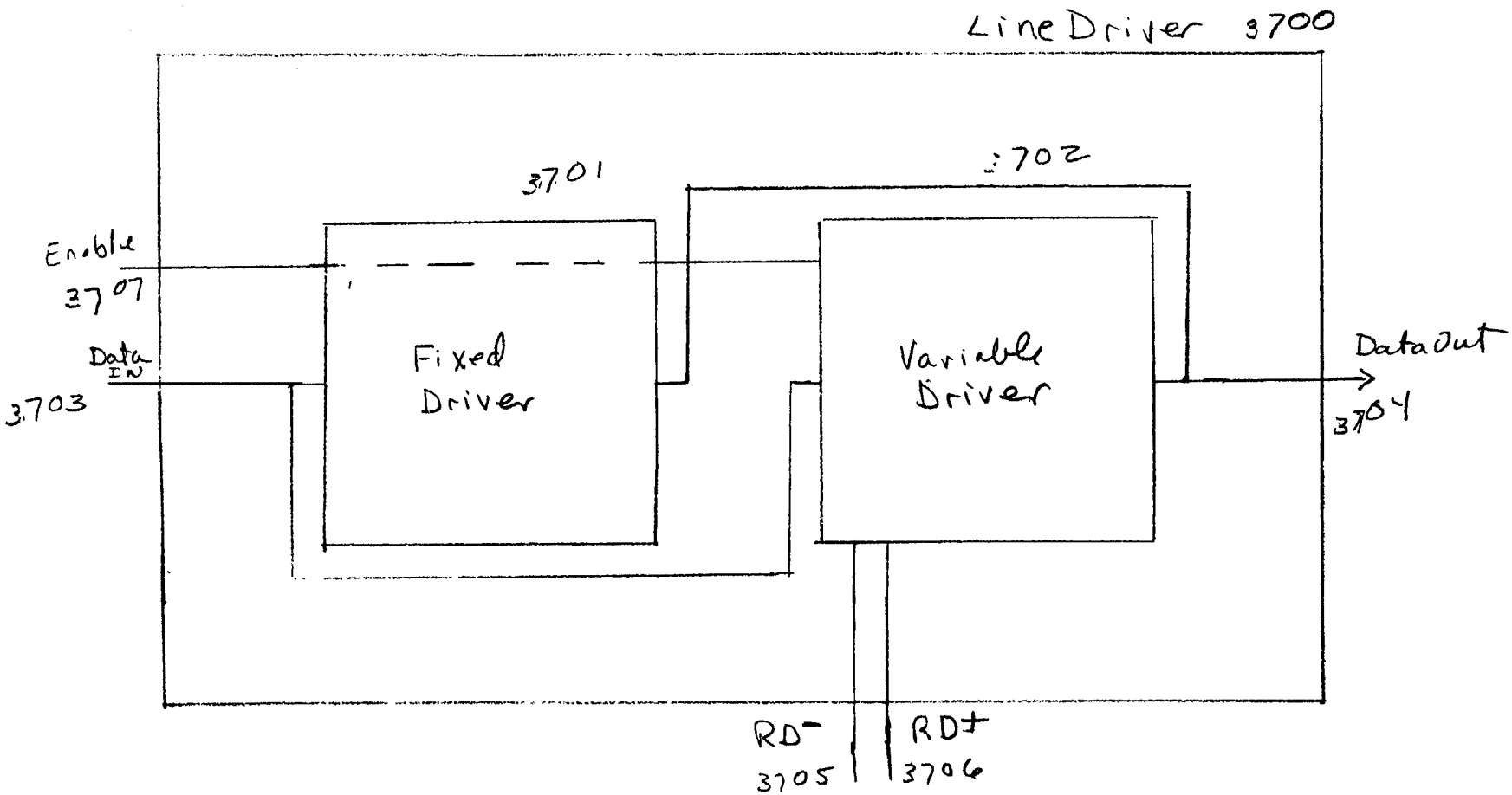

ROW
Addr
3630
ROW DEC
3601
3602
3611
3612
Section 0
Section 1
en0
en1
NXm
NXm
3650
Cofig
Port
CR
3604
3614
COL DEC &
COL$ 0
COL DEC &
COL$ 1
3603
3613
COL
Addr
2N

Fig 36



Variable Driver
- $RD^+ \wedge \overline{DataIN}$ = pull down
- $RD^- \wedge DataIN$ = pull up

Fig 37A

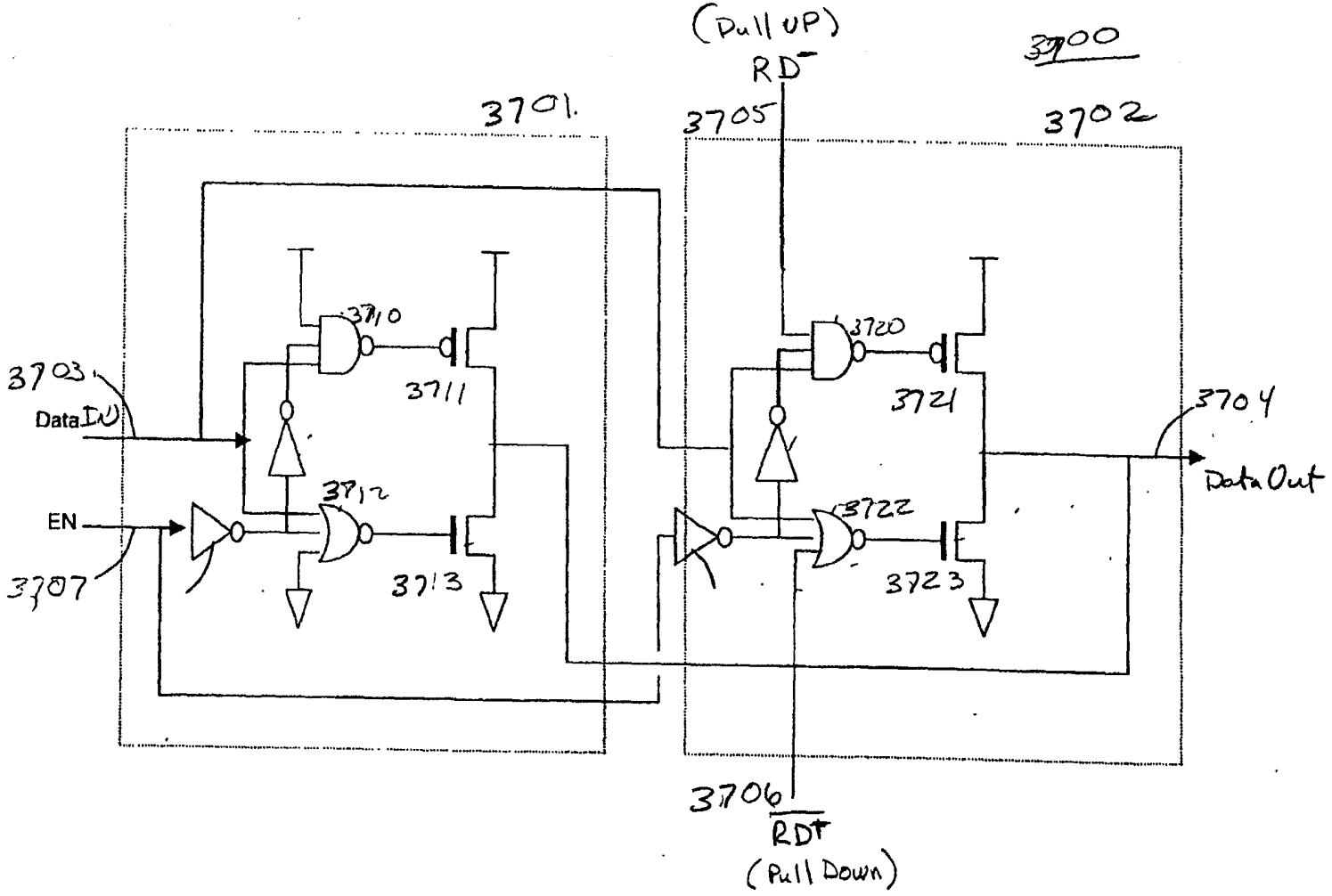


Fig 37B

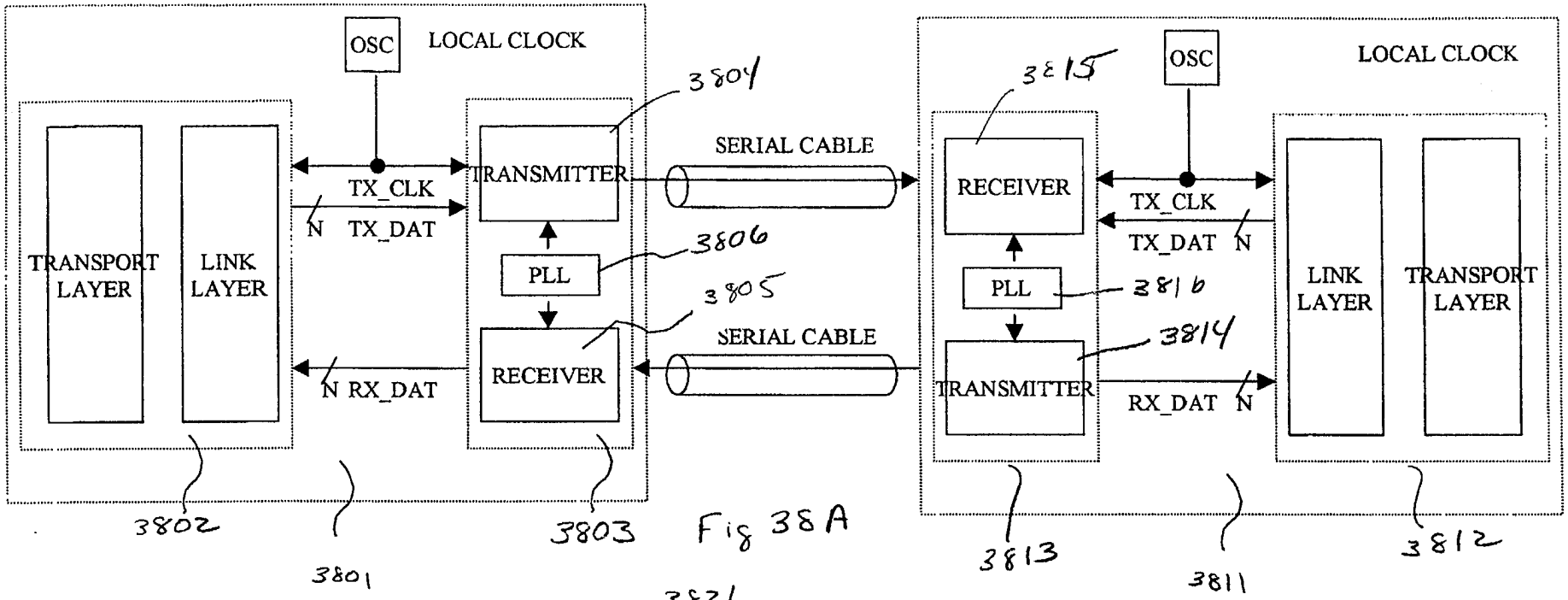


Fig 38A

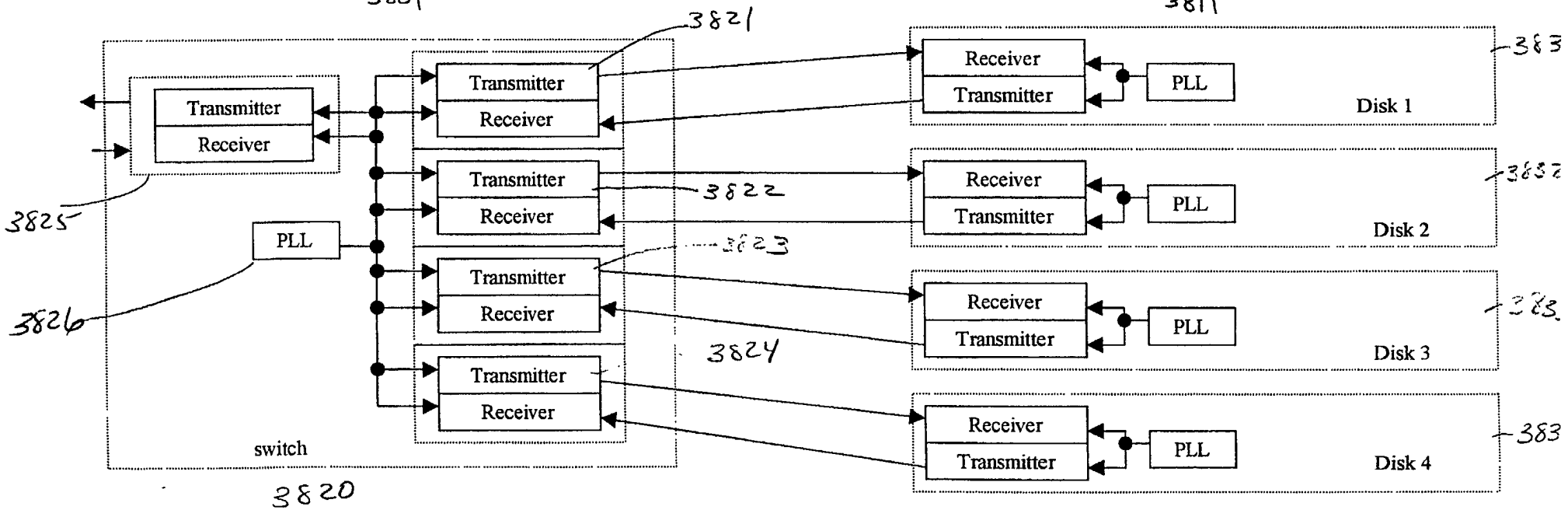


Fig 38B

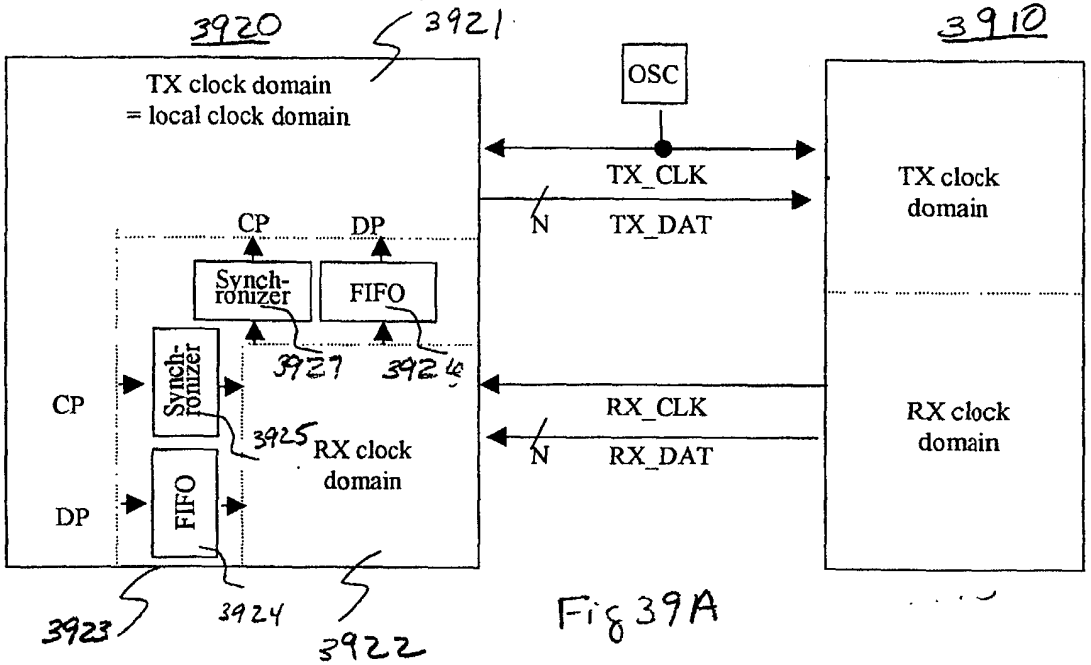


Fig 39A

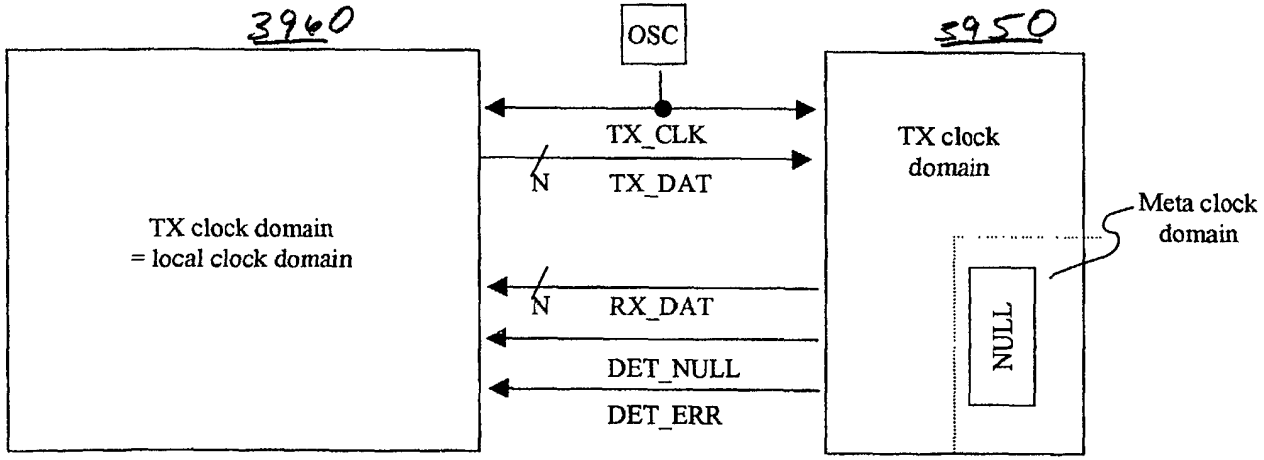


Fig 39B

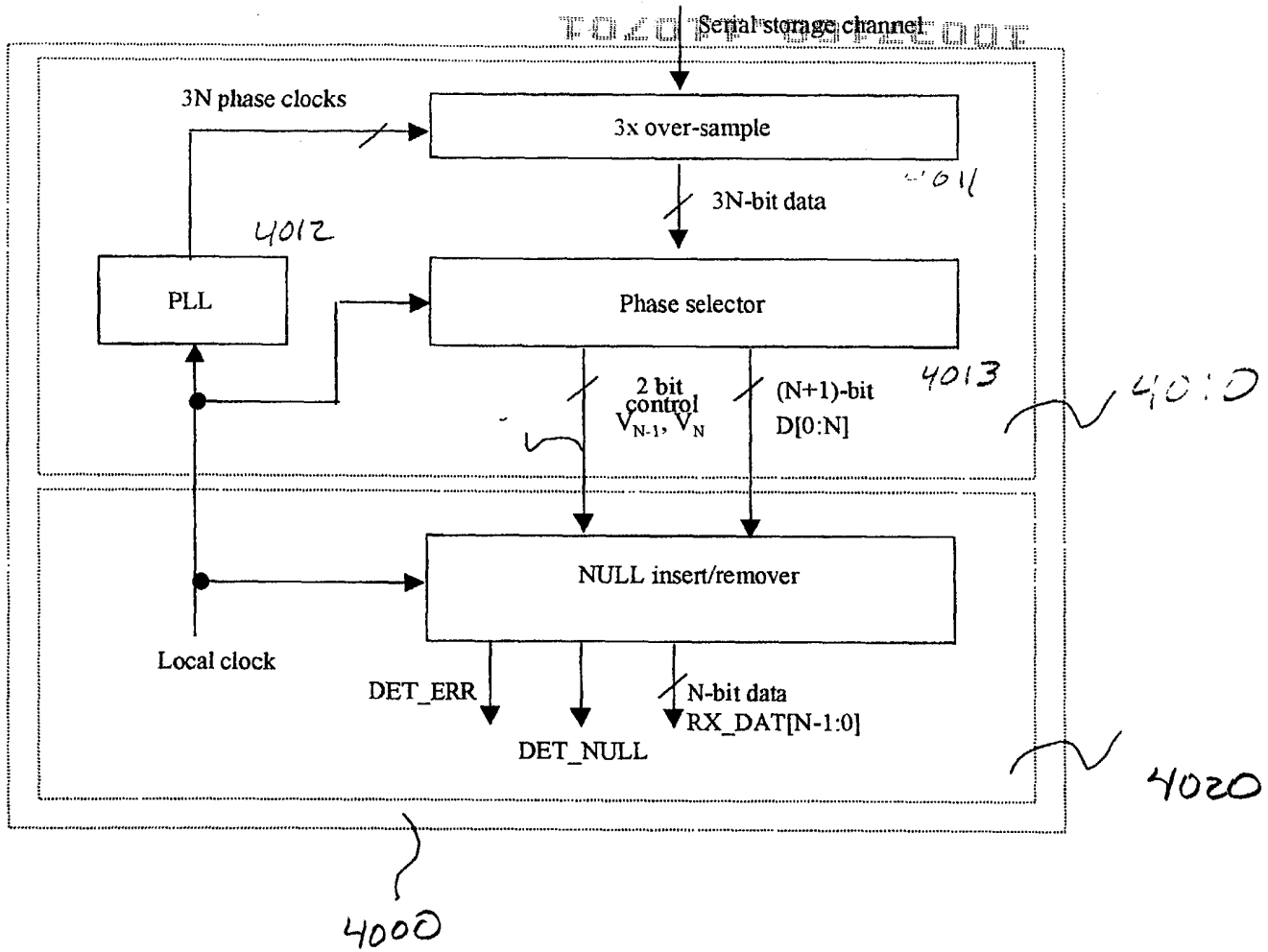

Serial storage channel
3N phase clocks
3x over-sample
4011
3N-bit data
4012
PLL
Phase selector
4013
2 bit control
V_{N-1}, V_N
(N+1)-bit
D[0:N]
4010
NULL insert/remover
Local clock
DET_ERR
DET_NULL
N-bit data
RX_DAT[N-1:0]
4020
4000

Fig 40

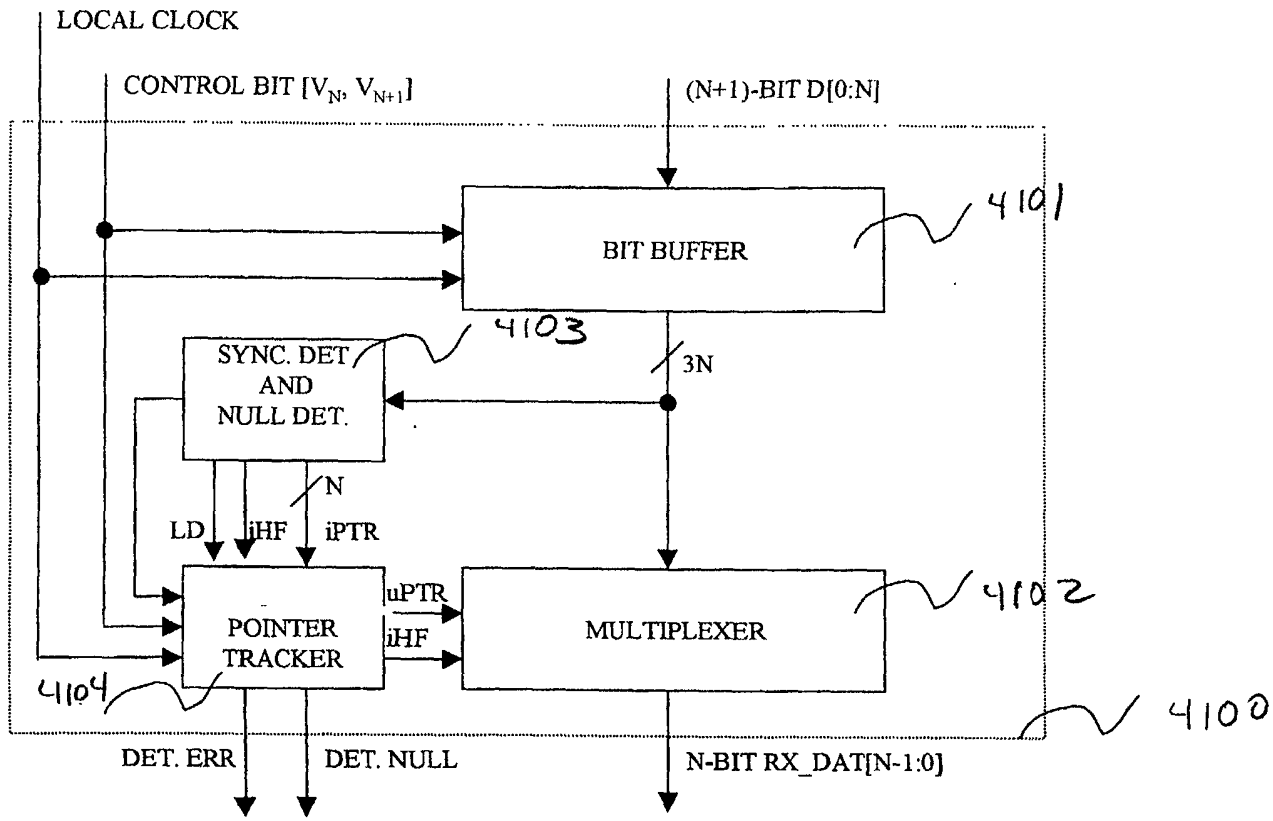

LOCAL CLOCK
CONTROL BIT [V_N, V_{N+1}]
(N+1)-BIT D[0:N]
BIT BUFFER
4101
4103
SYNC. DET AND NULL DET.
3N
N
LD
iHF
iPTR
POINTER TRACKER
uPTR
iHF
MULTIPLEXER
4102
4104
4100
DET. ERR
DET. NULL
N-BIT RX_DAT[N-1:0]

Fig 41

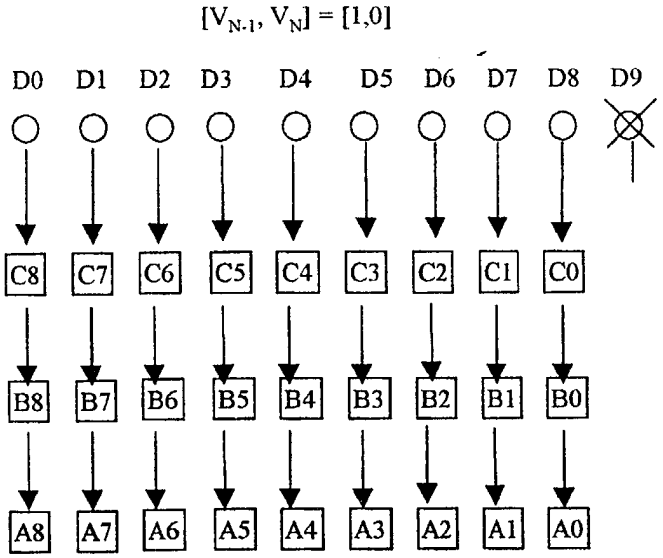


Fig 42A

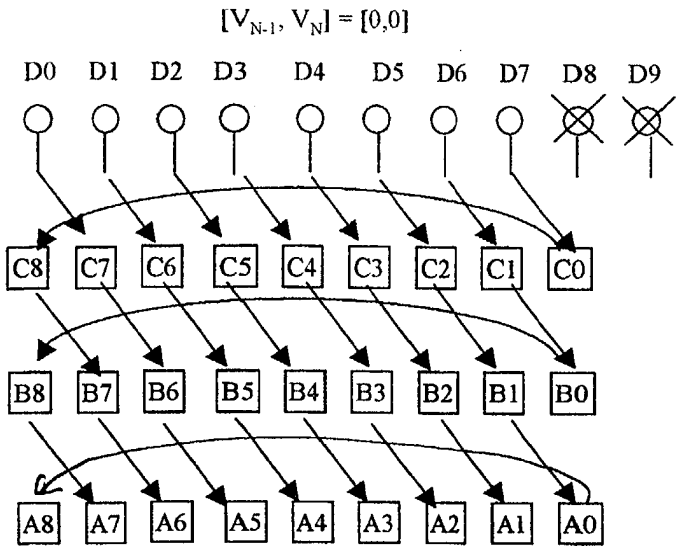


Fig 42B

$[V_{N-1}, V_N] = [1,1]$

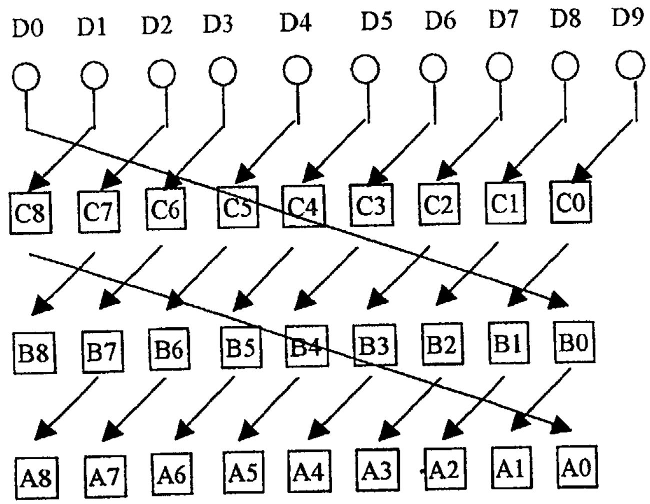


Fig 42C

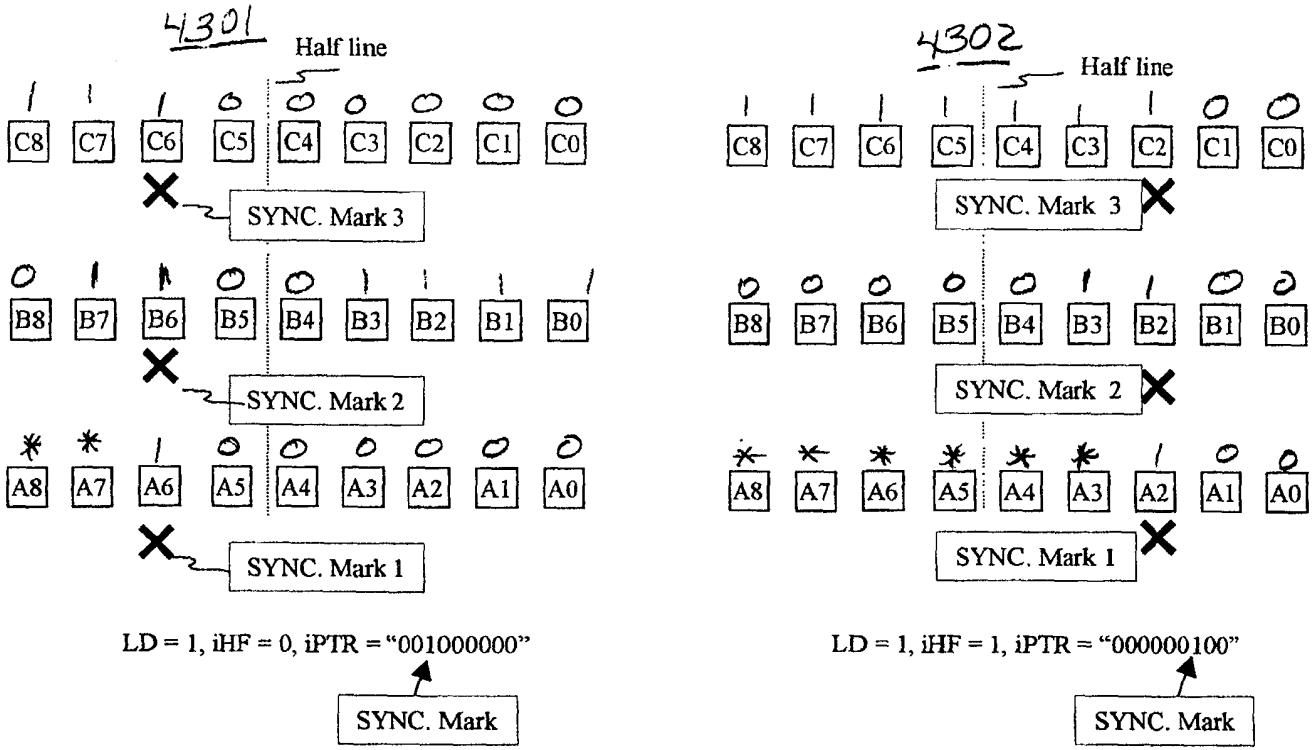
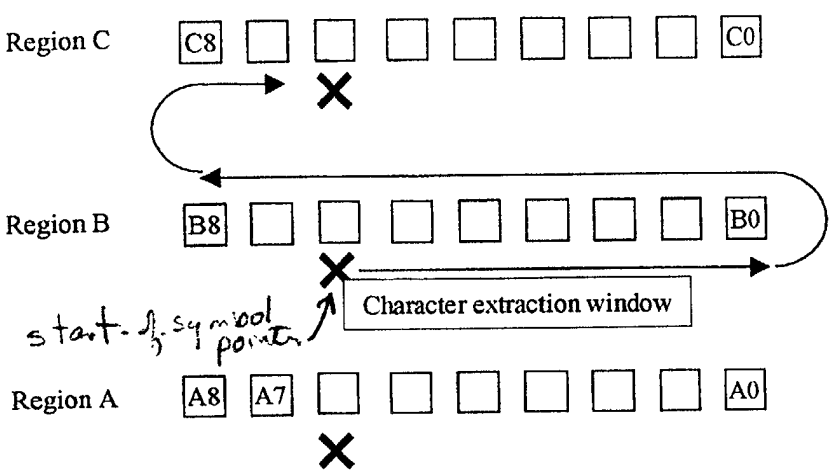
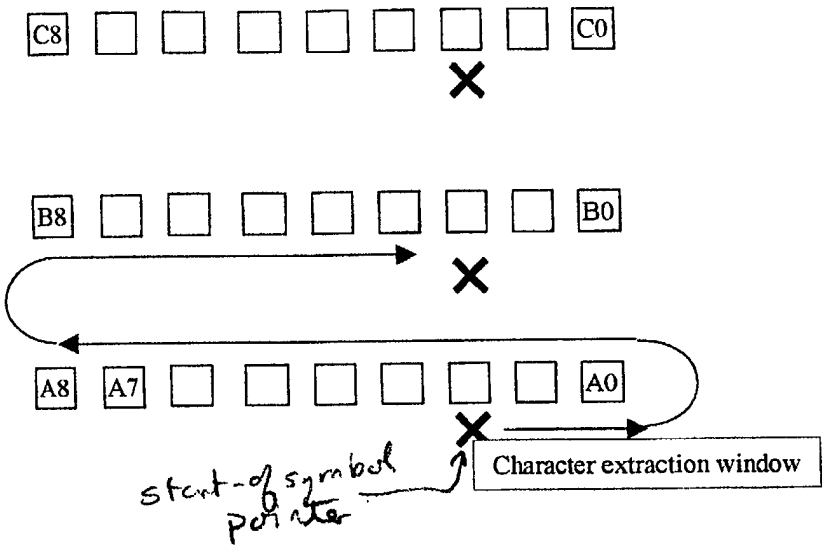


Fig. 43



LD = 1, iHF = 0, iPTR = "001000000"



LD = 1, iHF = 1, iPTR = "000000100"

Fig 44

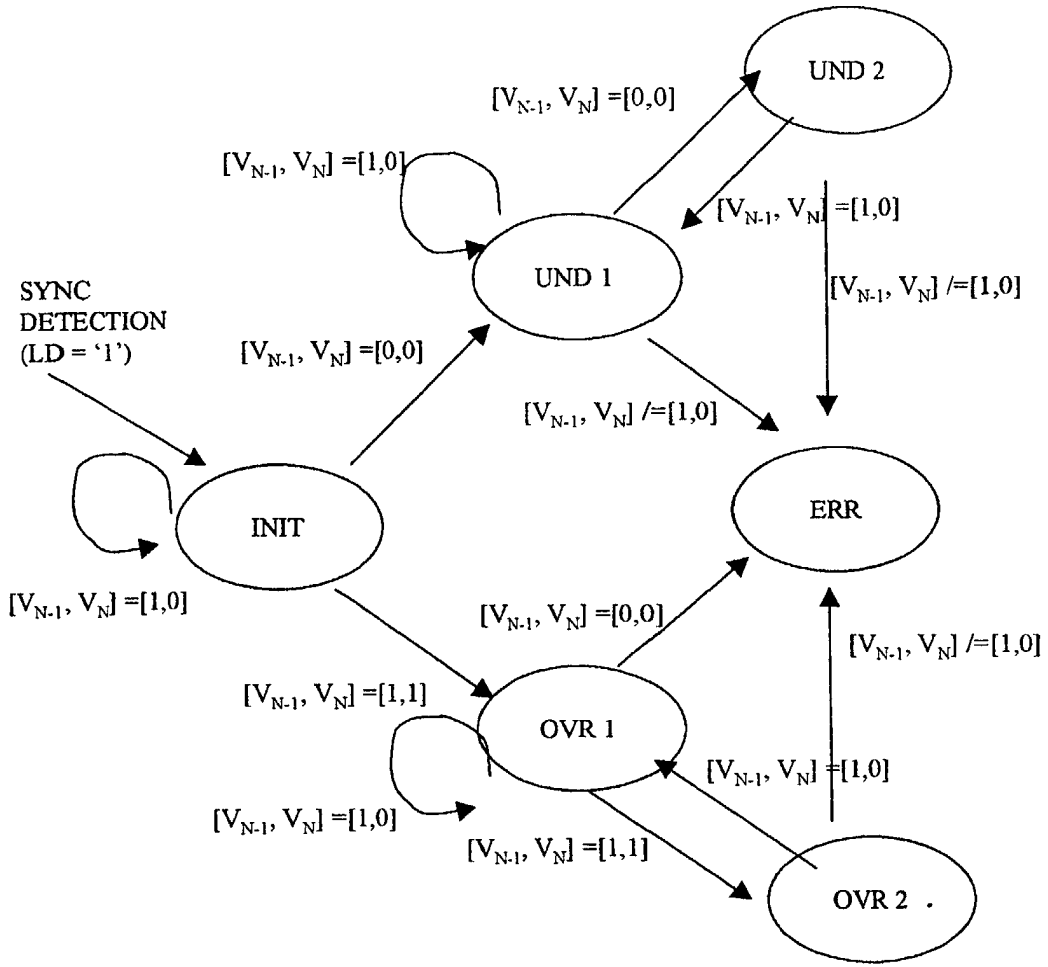

UND 2
$[V_{N-1}, V_N]$ =[0,0]
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ =[1,0]
UND 1
$[V_{N-1}, V_N]$ /=[1,0]
SYNC
DETECTION
(LD = '1')
$[V_{N-1}, V_N]$ =[0,0]
$[V_{N-1}, V_N]$ /=[1,0]
ERR
INIT
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ =[0,0]
$[V_{N-1}, V_N]$ /=[1,0]
$[V_{N-1}, V_N]$ =[1,1]
OVR 1
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ =[1,1]
OVR 2

Fig 45

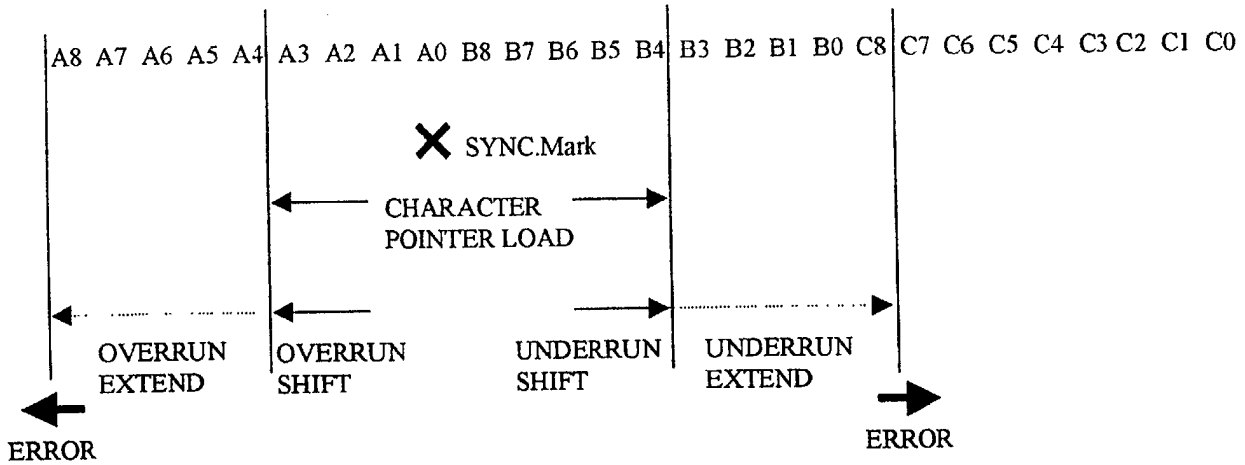

A8 A7 A6 A5 A4 A3 A2 A1 A0 B8 B7 B6 B5 B4 B3 B2 B1 B0 C8 C7 C6 C5 C4 C3 C2 C1 C0
SYNC.Mark
CHARACTER
POINTER LOAD
OVERRUN
EXTEND
OVERRUN
SHIFT
UNDERRUN
SHIFT
UNDERRUN
EXTEND
ERROR
ERROR

Fig 46

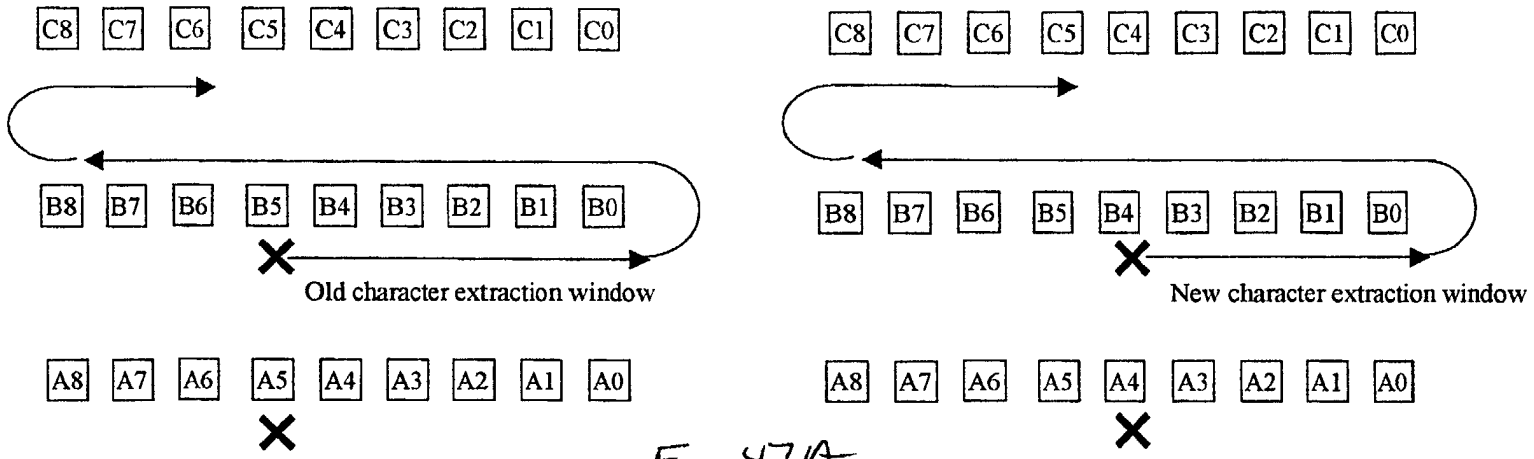


Fig 47A

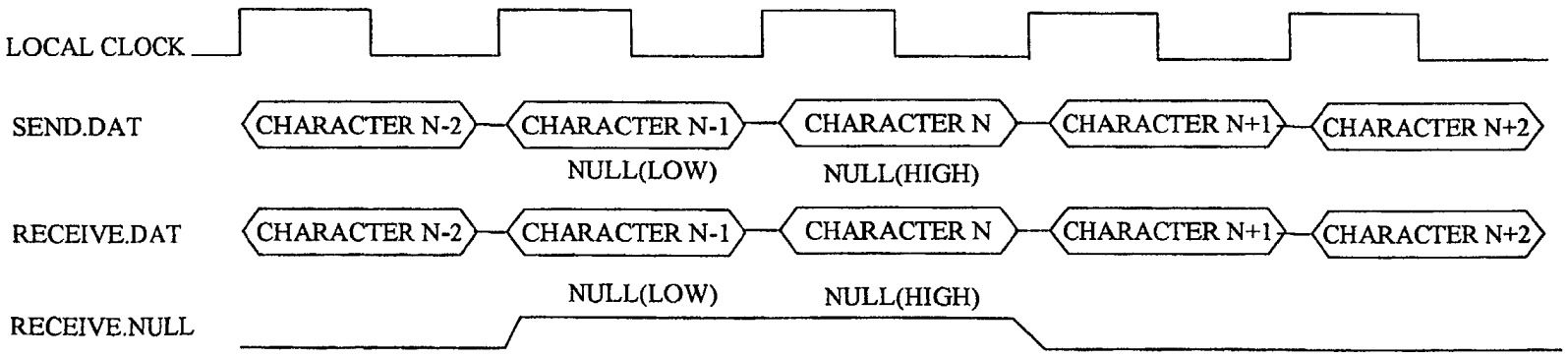


Fig 47B

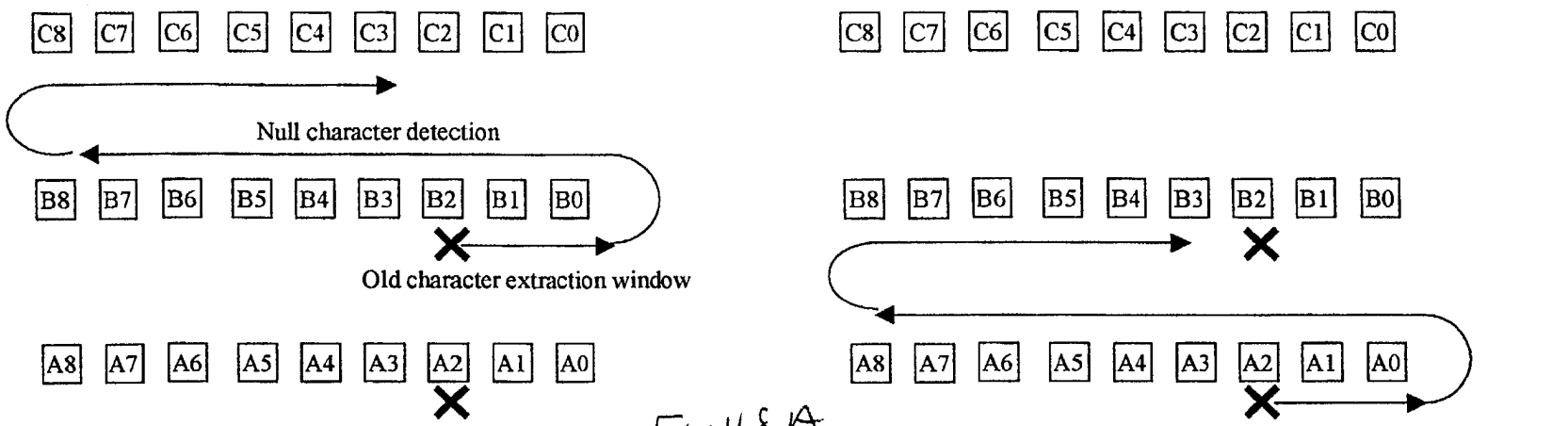


Fig 48A

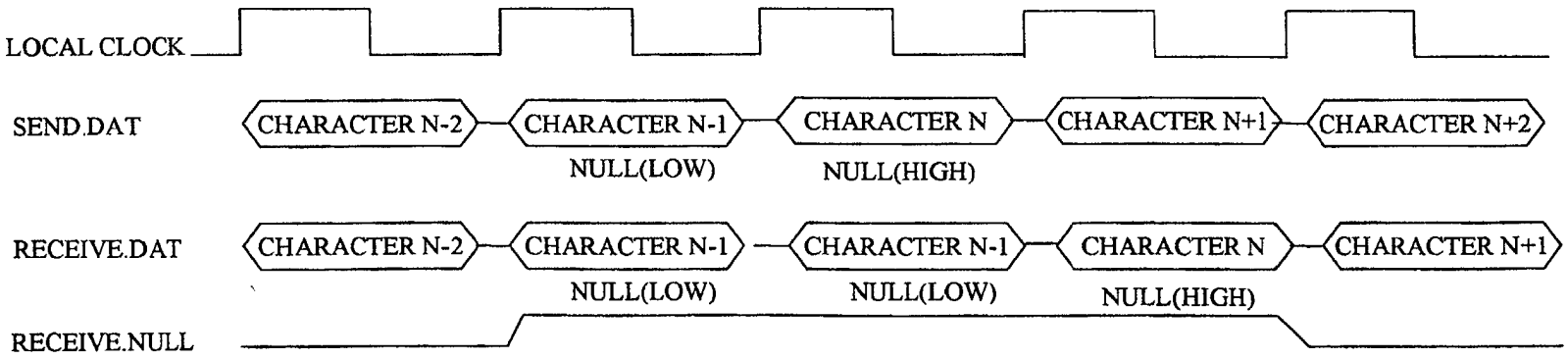


Fig 48B

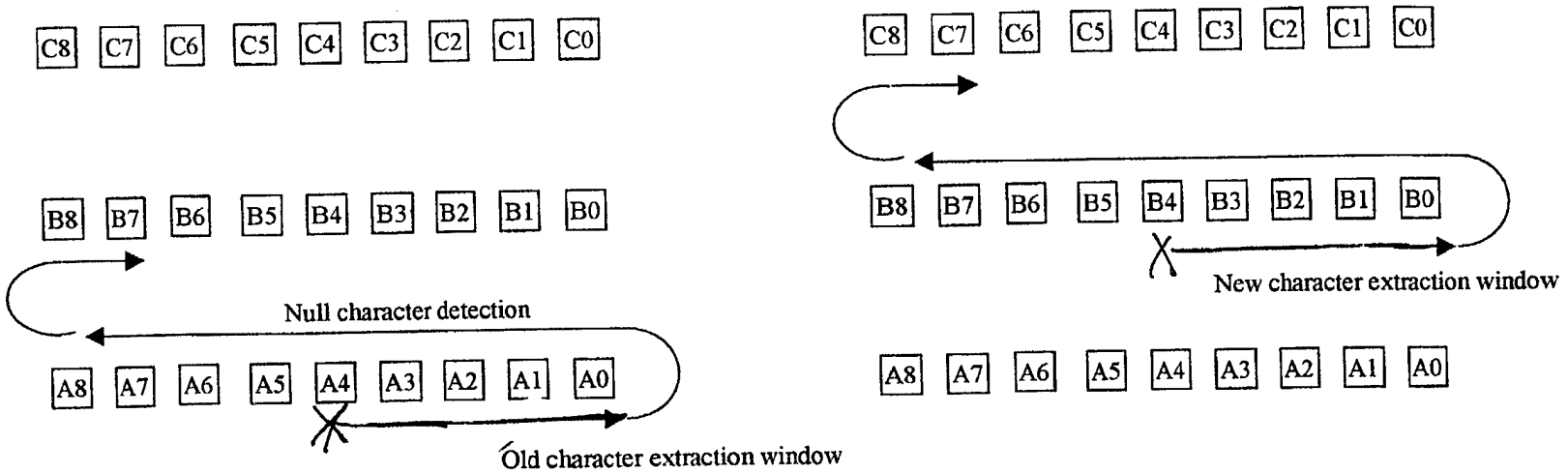


Fig. 49A

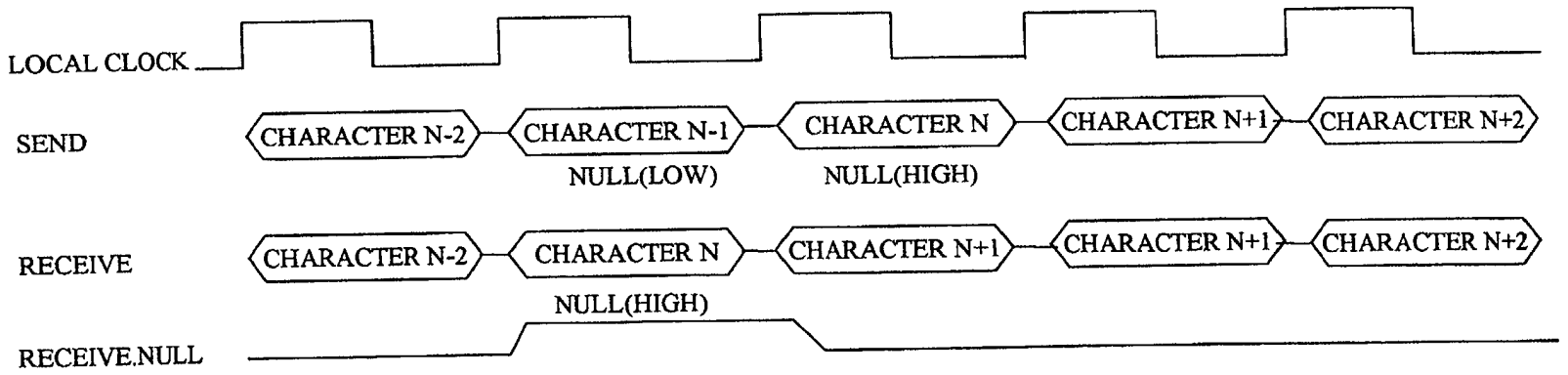


Fig 49B